Self-Adjusting Datacenter Networks for the AI/ML Era

Stefan Schmid (TU Berlin)

“We cannot direct the wind,
but we can adjust the sails.”
(Folklore)

Acknowledgements:

GIF German-Israeli Foundation for Scientific Research and Development

erc European Research Council Established by the European Commission

We live in

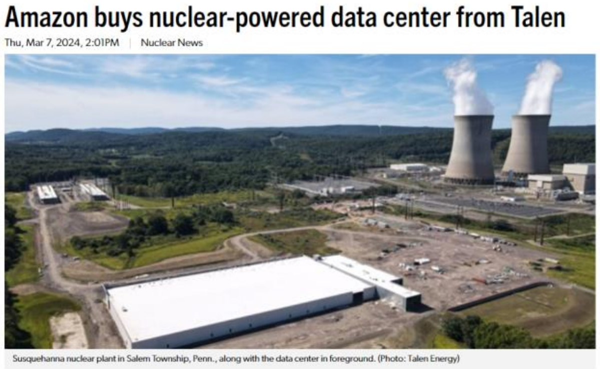
The Age of Computation

Datacenters (“hyper-scale”)

Data intensive applications requiring significant processing.

We live in

The Age of Computation

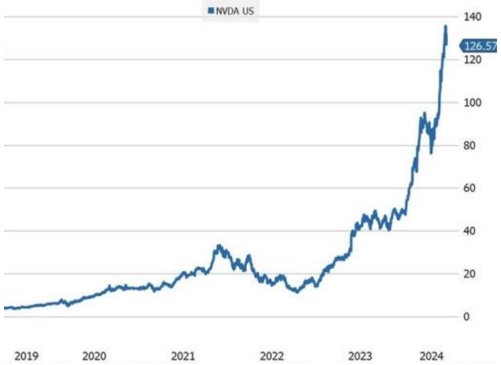


Amazon buys nuclear-powered data center from Talen

Thu, Mar 7, 2024, 2:01PM | Nuclear News

Susquehanna nuclear plant in Salem Township, Penn., along with the data center in foreground. (Photo: Talen Energy)

Training even across ***multiple datacenters*** (and ***powerplants***)!

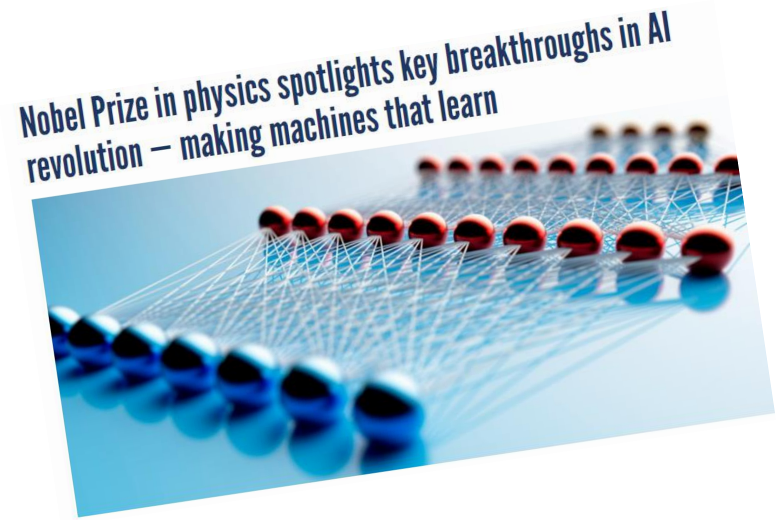


Nvidia: fastest growing company ever

Energy consumption and probably also computation trends will likely stay. ***Kardashev Scale*** even classifies civilizations by their energy use!

We live in

The Age of Computation



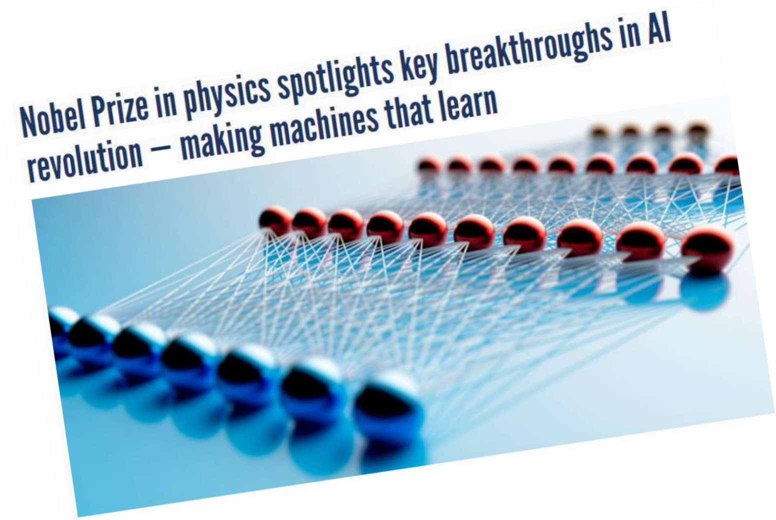
Nobel Prize in physics spotlights key breakthroughs in AI revolution – making machines that learn

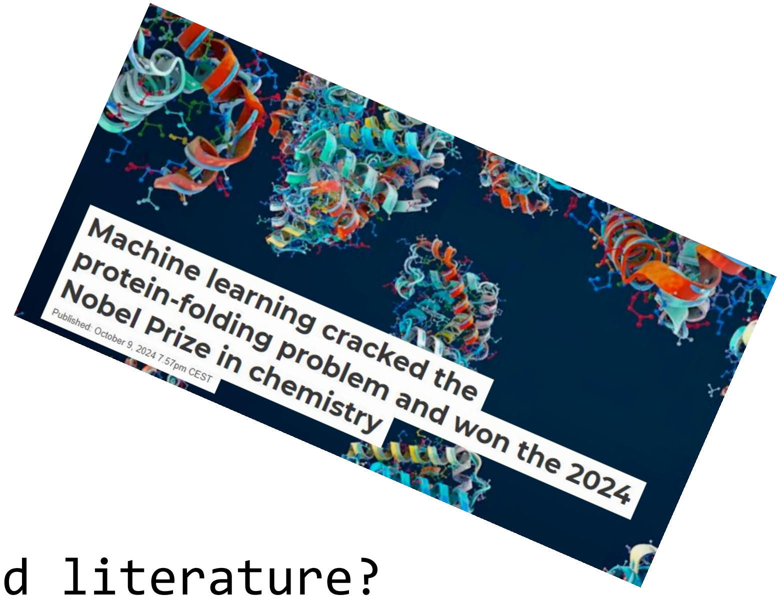
Machine learning cracked the protein-folding problem and won the 2024 Nobel Prize in chemistry

Published: October 9, 2024 7.57pm CEST

We live in

The Age of Computation


Nobel Prize in physics spotlights key breakthroughs in AI revolution – making machines that learn


Machine learning cracked the protein-folding problem and won the 2024 Nobel Prize in chemistry

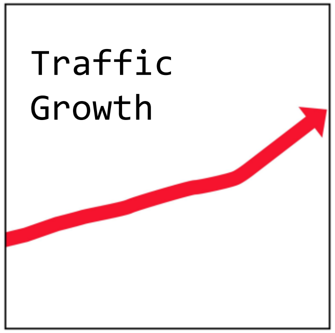
Published: October 9, 2024 7.57pm CEST

… soon in economics and literature?

Networks Matter!

Distributed Applications Require Networks

+network



Source: Facebook

Interconnecting networks:
a **critical infrastructure**
of our digital society.

Networks Matter!

Distributed Applications Require Networks

Interconnecting networks:
a **critical infrastructure**
of our digital society.

Credits: Marco Chiesa

The Problem

Huge Infrastructure, Inefficient Use

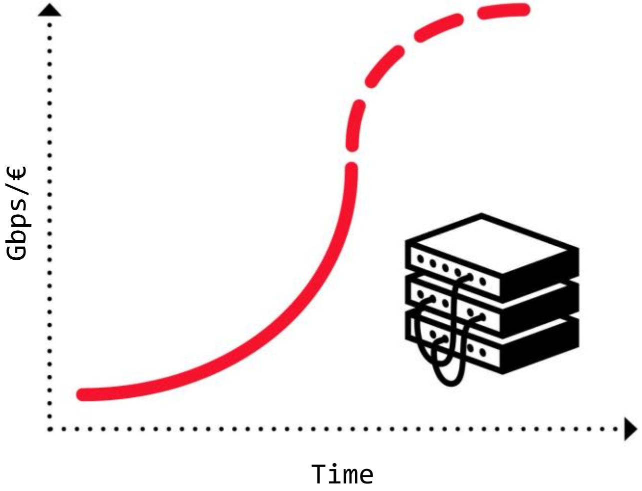
⇢ Network equipment reaching capacity limits

⇀ Transistor density rates stalling

⇀ “End of **Moore‘s Law** in networking”

⇢ Hence: more equipment, larger networks

⇢ Resource intensive and: **inefficient**



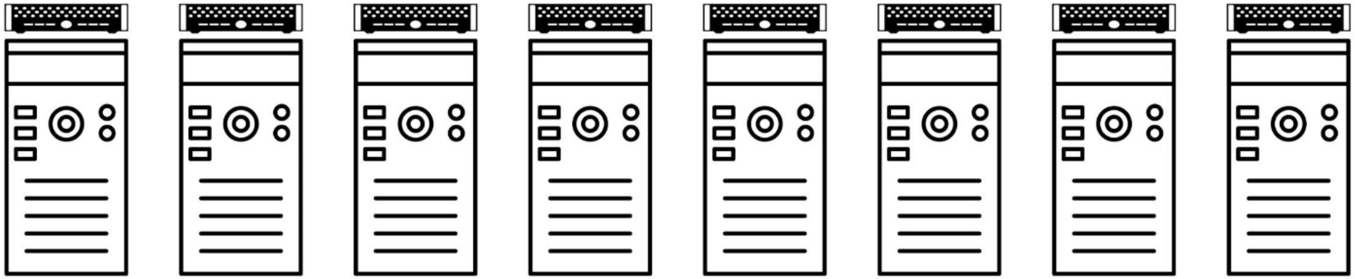
[1] Source: Microsoft, 2019

Annoying for companies, **opportunity** for researchers!

Root Cause

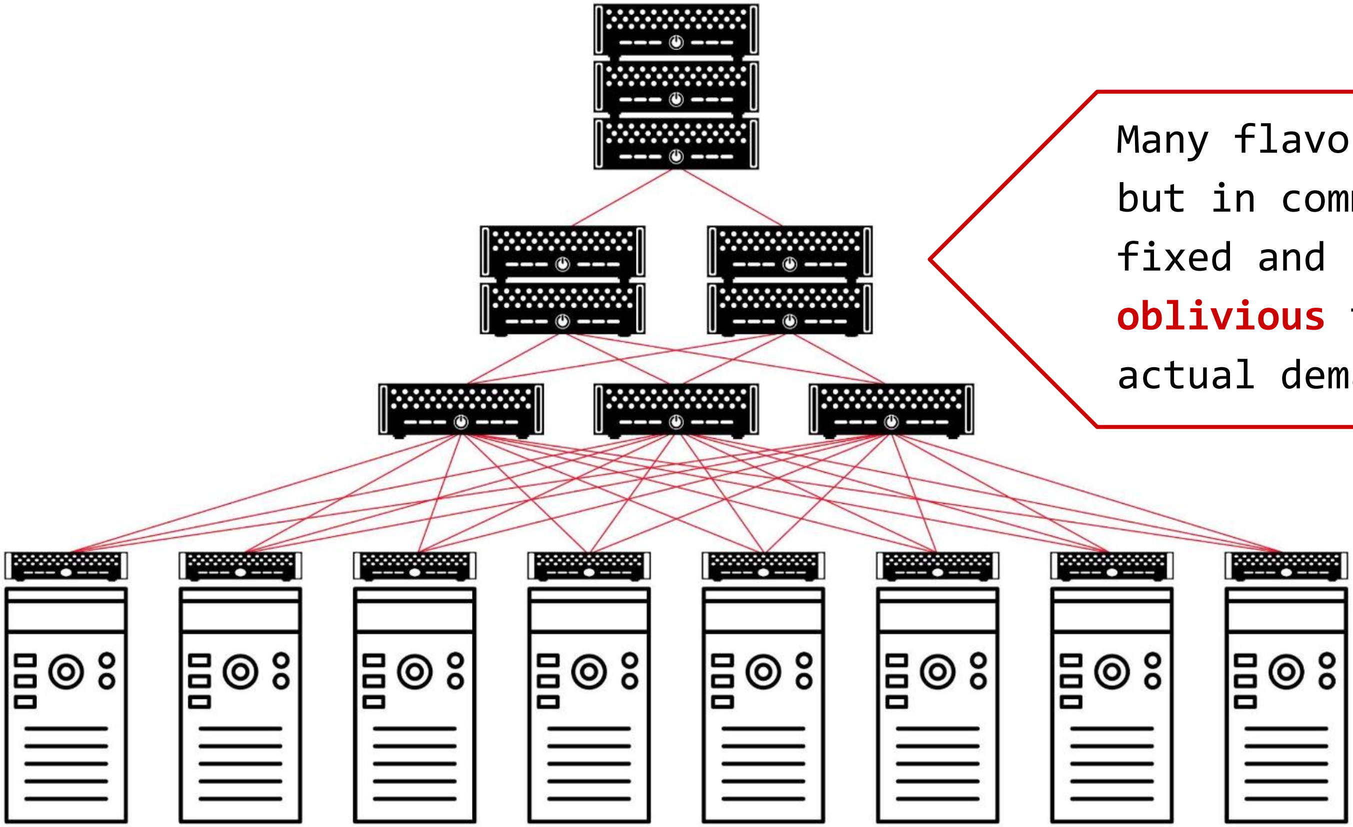
Fixed and Demand-Oblivious Topology

How to interconnect?



Root Cause

Fixed and Demand-Oblivious Topology

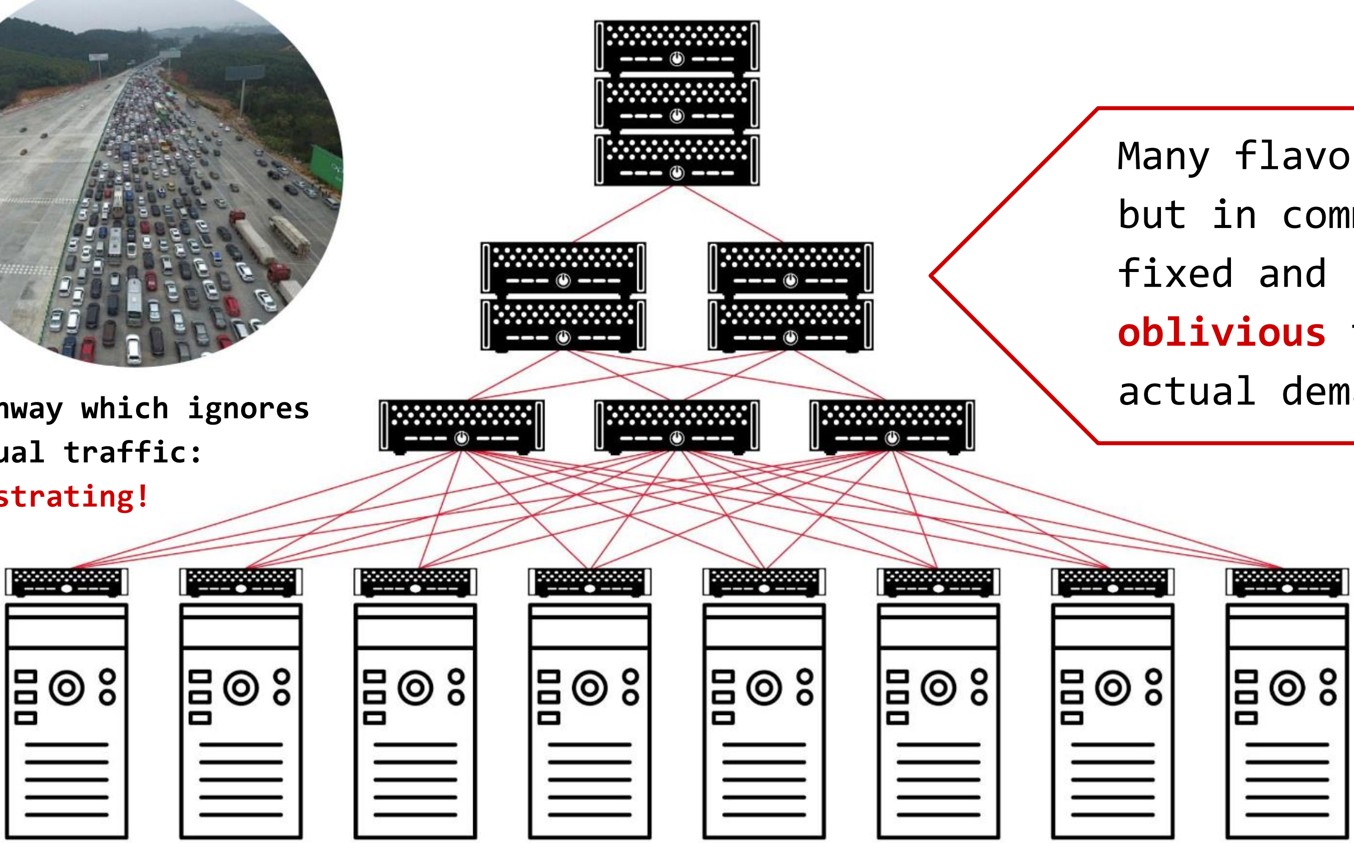


Many flavors, but in common: fixed and **oblivious** to actual demand.

Root Cause

Fixed and Demand-Oblivious Topology

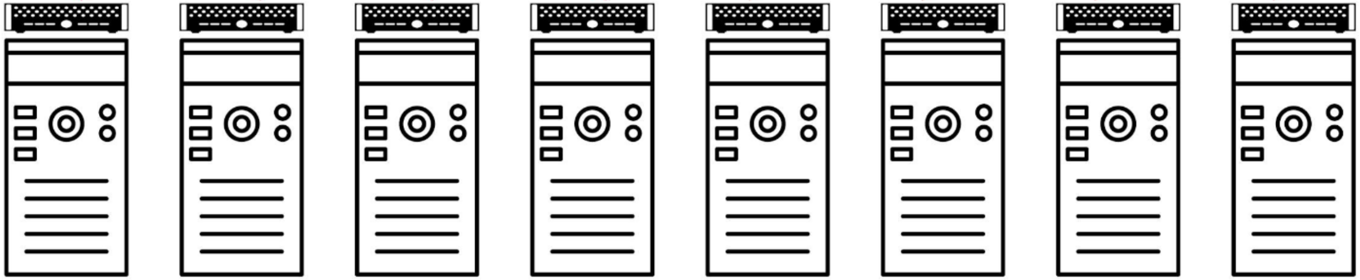
Highway which ignores actual traffic: frustrating!



Many flavors, but in common: fixed and **oblivious** to actual demand.

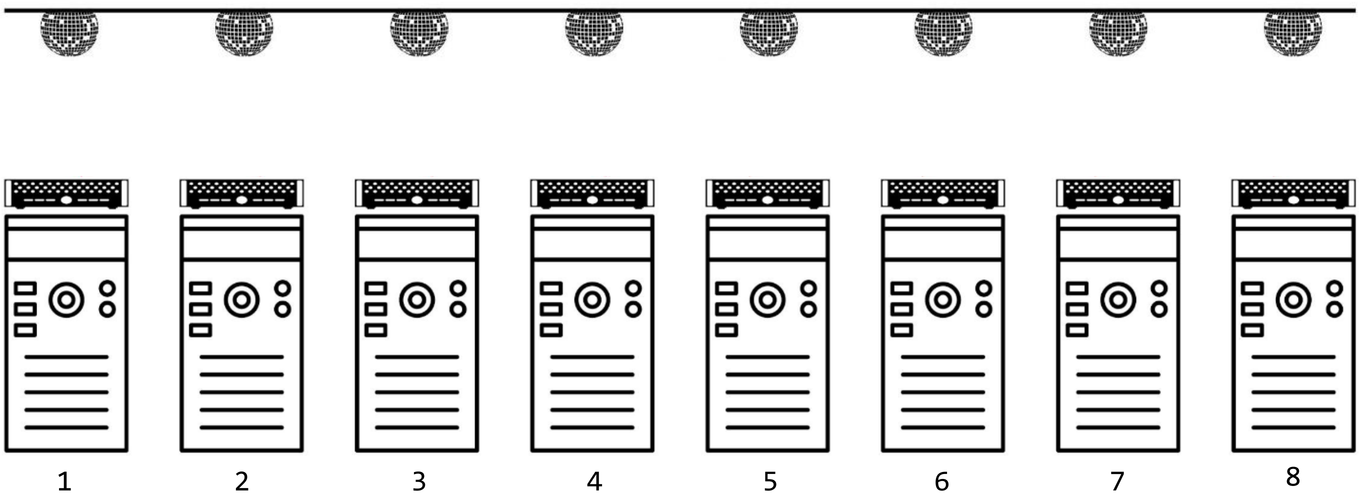
A Vision

Flexible and Demand-Aware Topologies



A Vision

Flexible and Demand-Aware Topologies



e.g., mirrors

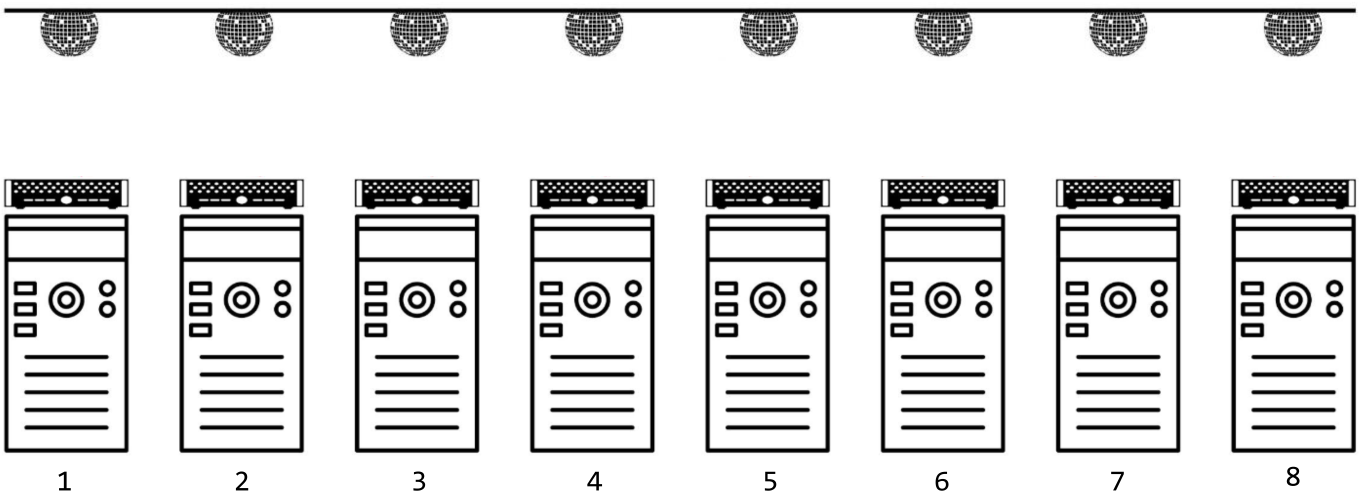
new flexible interconnect

A Vision

Flexible and Demand-Aware Topologies

demand matrix:

	1	2	3	4	5	6	7	8
1					🟥			
2						🟥		
3							🟥	
4								🟥
5	🟥							
6		🟥						
7			🟥					
8				🟥				



e.g., mirrors

new flexible interconnect

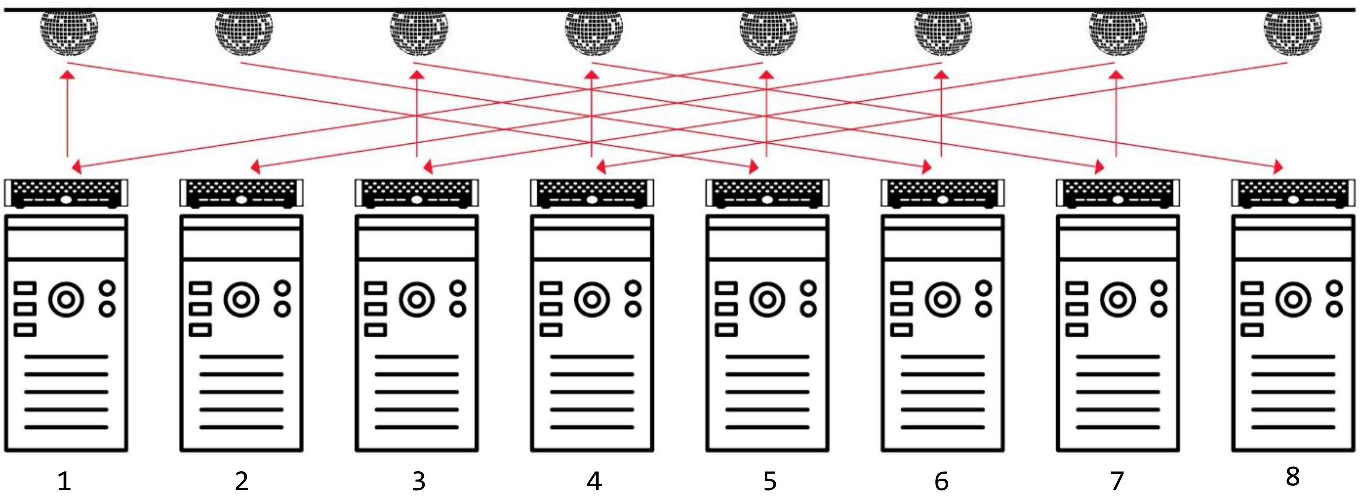
A Vision

Flexible and Demand-Aware Topologies

Matches demand

demand matrix:

	1	2	3	4	5	6	7	8
1					🟥			
2						🟥		
3							🟥	
4								🟥
5	🟥							
6		🟥						
7			🟥					
8				🟥				



e.g., mirrors

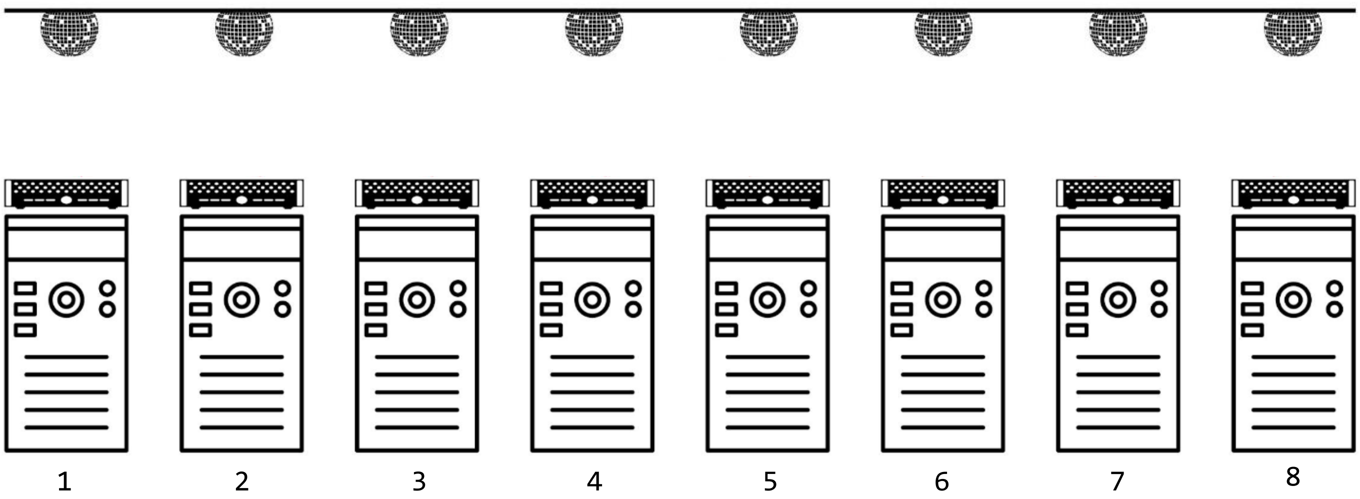
new flexible interconnect

A Vision

Flexible and Demand-Aware Topologies

new demand:

	1	2	3	4	5	6	7	8
1		🟥						
2	🟥							
3				🟥				
4			🟥					
5						🟥		
6					🟥			
7								🟥
8							🟥	



e.g., mirrors

new flexible interconnect

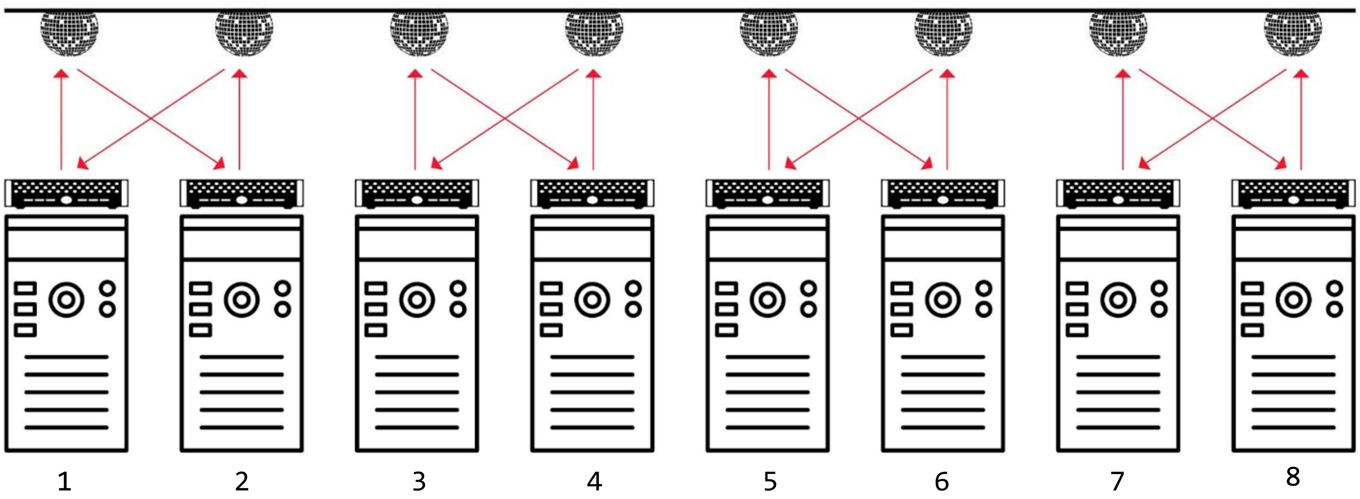
A Vision

Flexible and Demand-Aware Topologies

Matches demand

new demand:

	1	2	3	4	5	6	7	8
1		🟥						
2	🟥							
3				🟥				
4			🟥					
5						🟥		
6					🟥			
7								🟥
8							🟥	



e.g., mirrors

new flexible interconnect

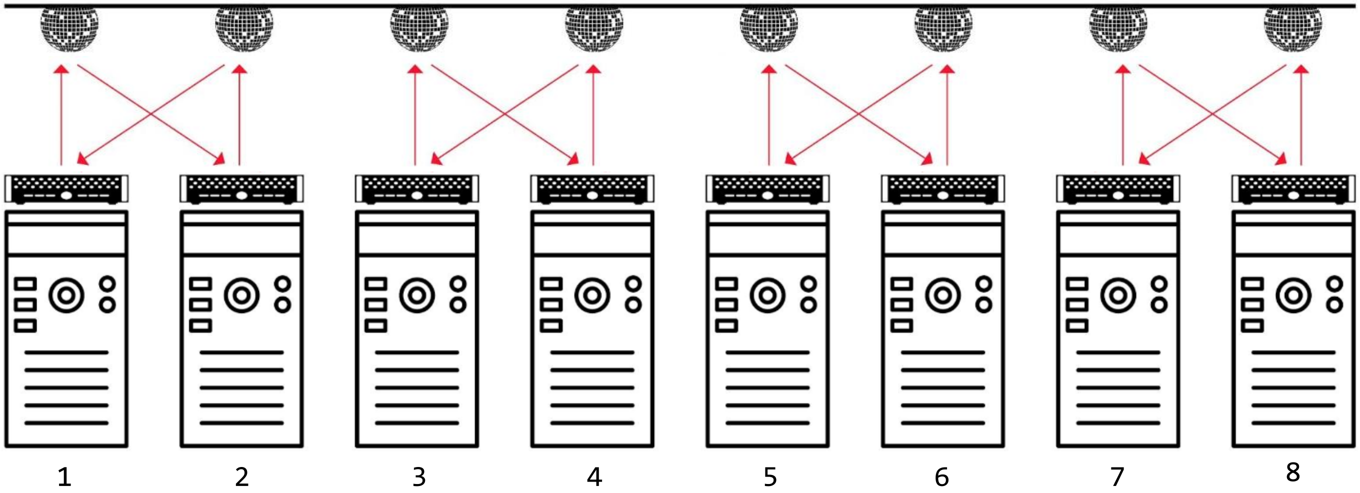
A Vision

Flexible and Demand-Aware Topologies

Self-Adjusting Networks

new demand:

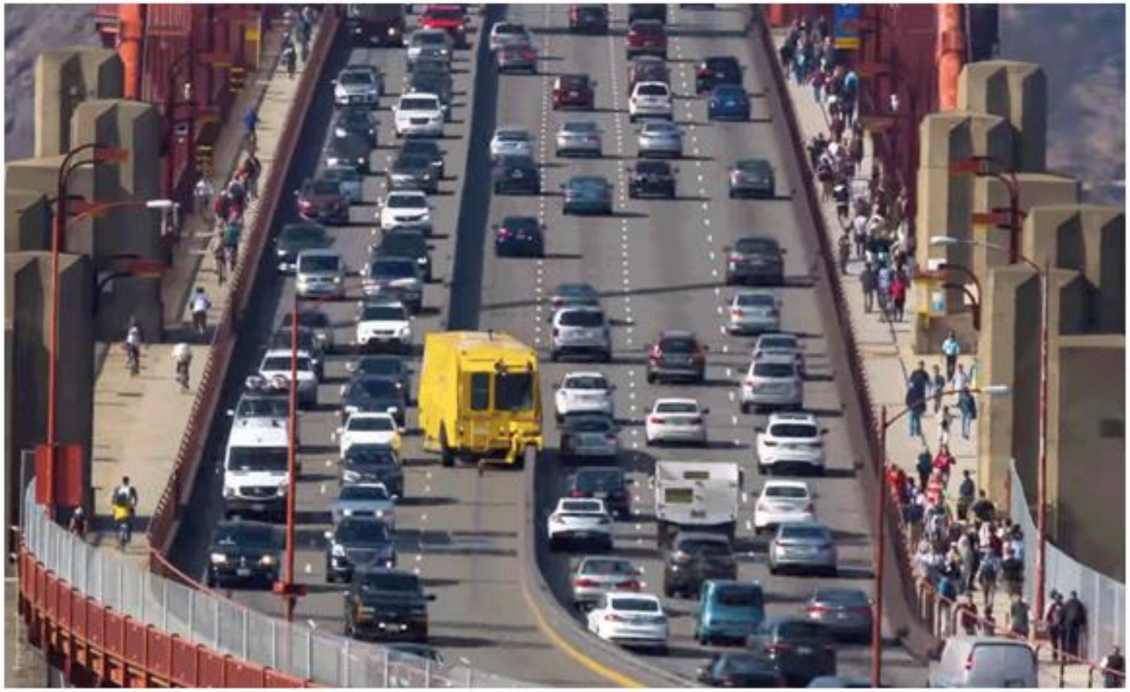
	1	2	3	4	5	6	7	8
1		🟥						
2	🟥							
3				🟥				
4			🟥					
5						🟥		
6					🟥			
7								🟥
8							🟥	



e.g., mirrors

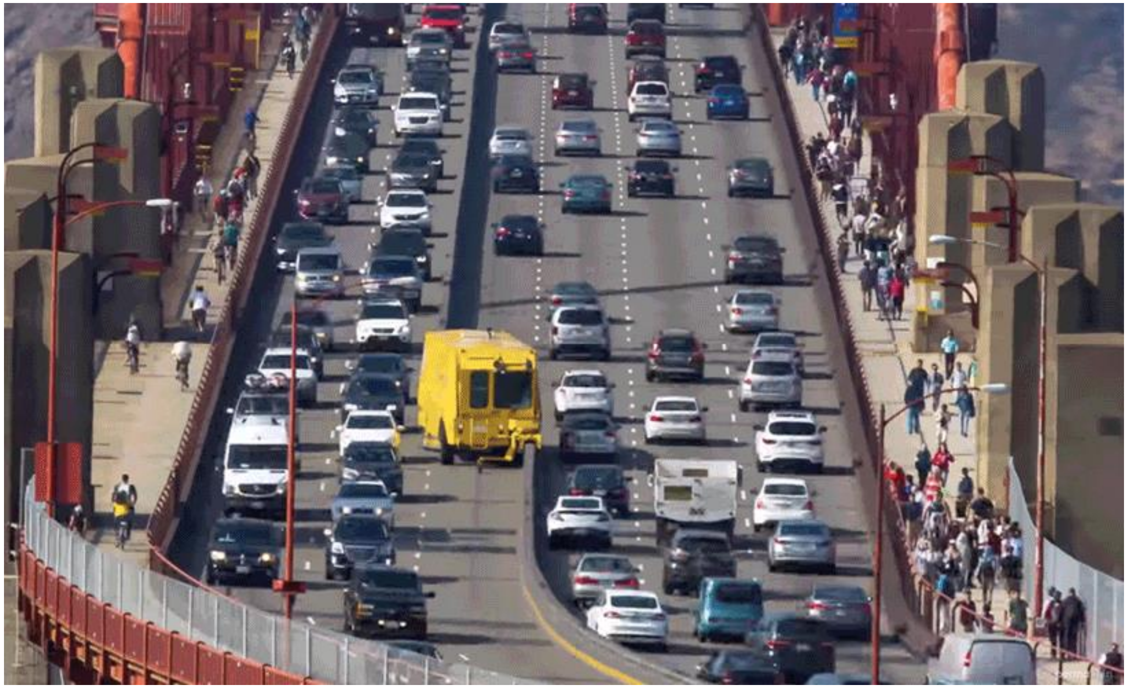
new flexible interconnect

Analogy



Golden Gate Zipper

Analogy



Golden Gate Zipper

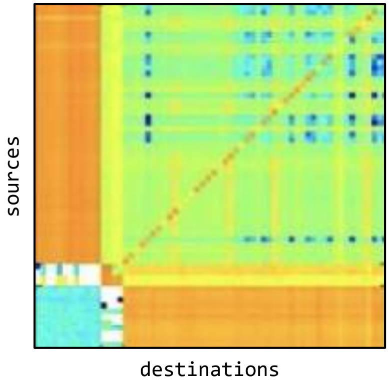
The Motivation

Much Structure in the Demand

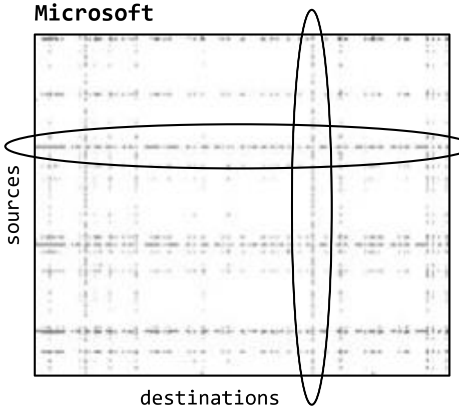
Empirical studies:

traffic matrices sparse and skewed

Facebook

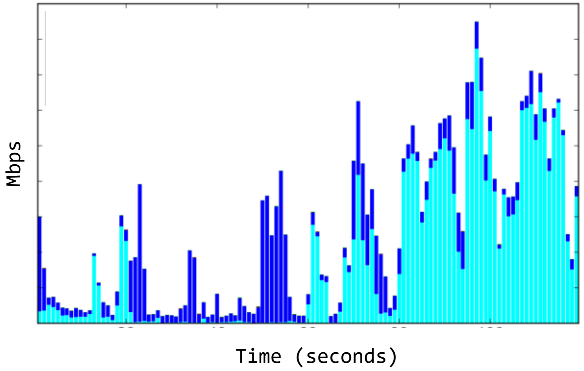


Microsoft



traffic bursty over time

Facebook



The hypothesis: can be exploited.

Recent Representation of Trace Structure:

Complexity Map

● DB

● Web

● Had

● ML

● CNS

● Multi Grid

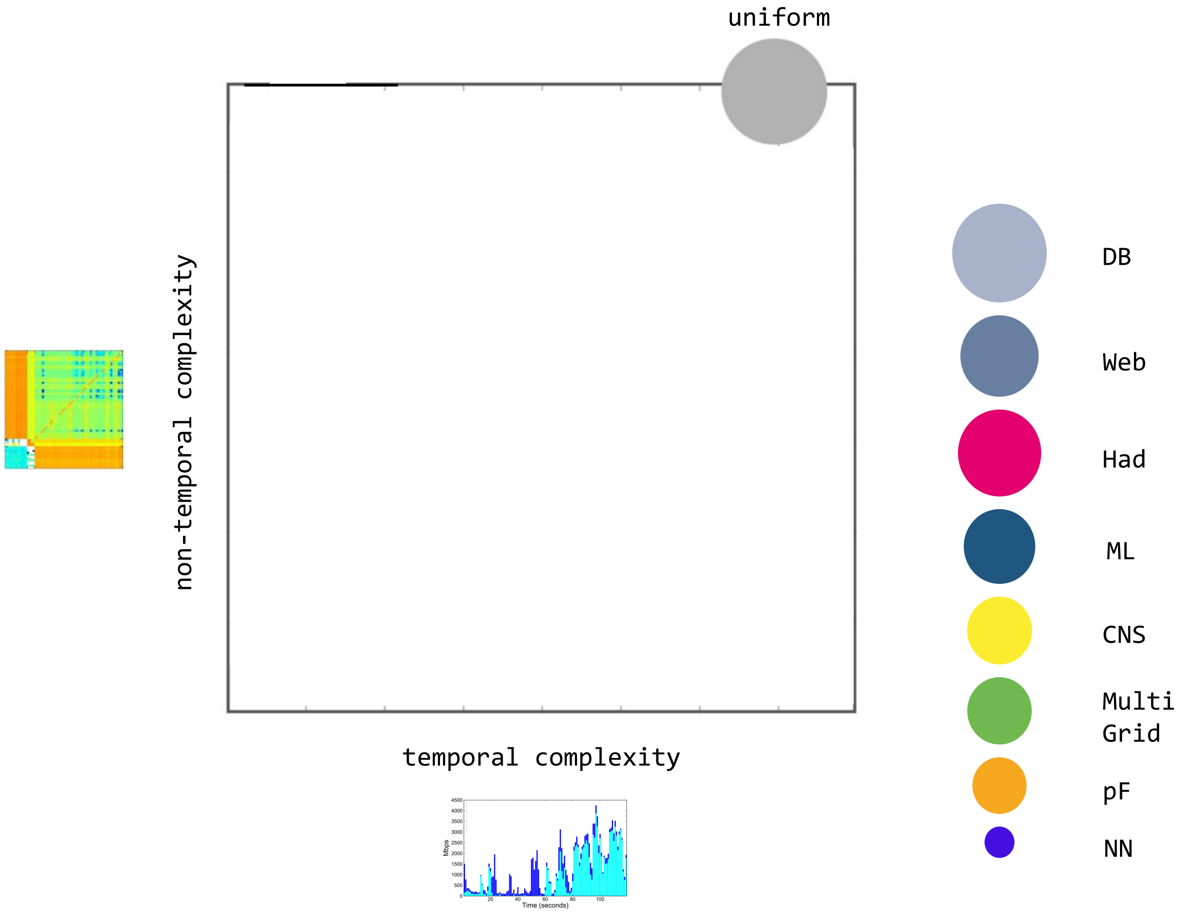
● pF

● NN

Griner et al., SIGMETRICS 2020

Recent Representation of Trace Structure:

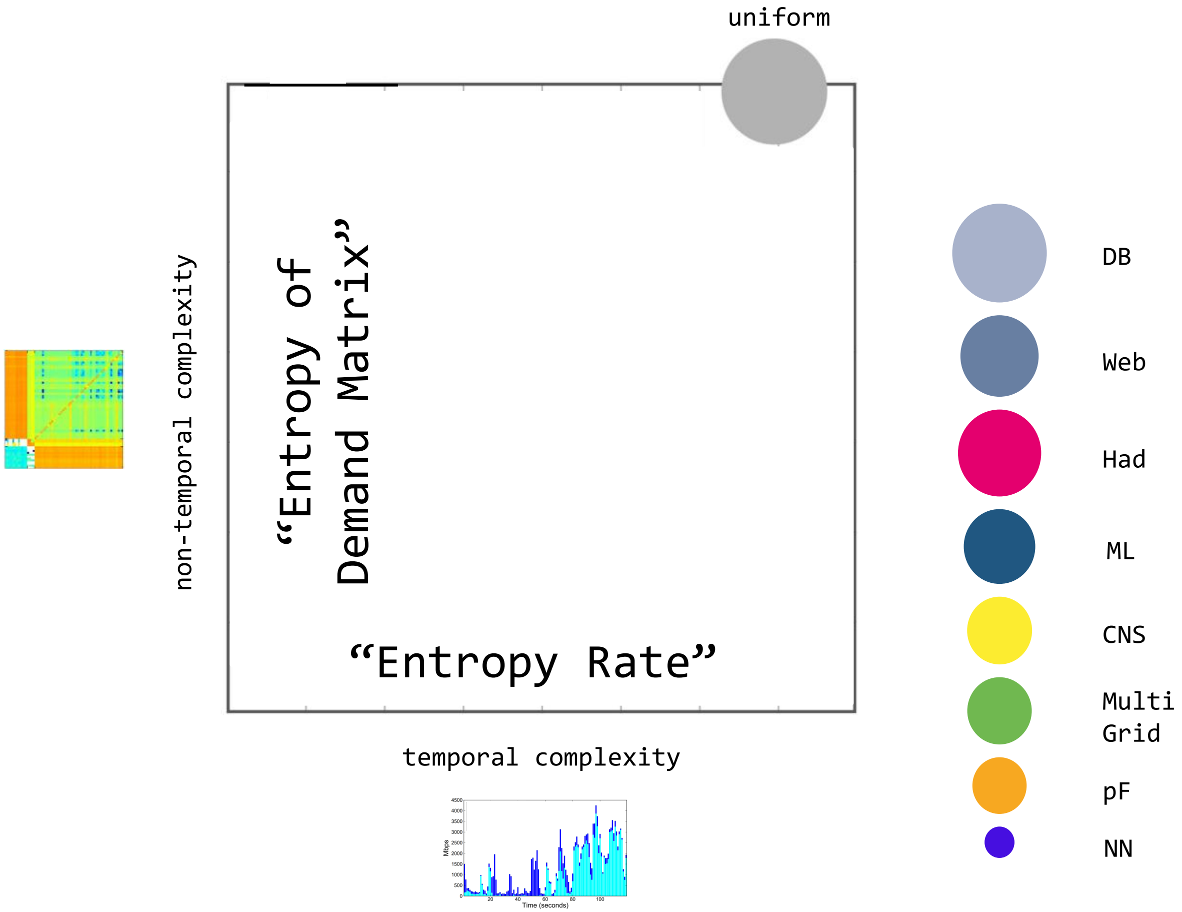
Complexity Map



Griner et al., SIGMETRICS 2020

Recent Representation of Trace Structure:

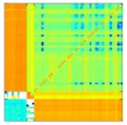
Complexity Map

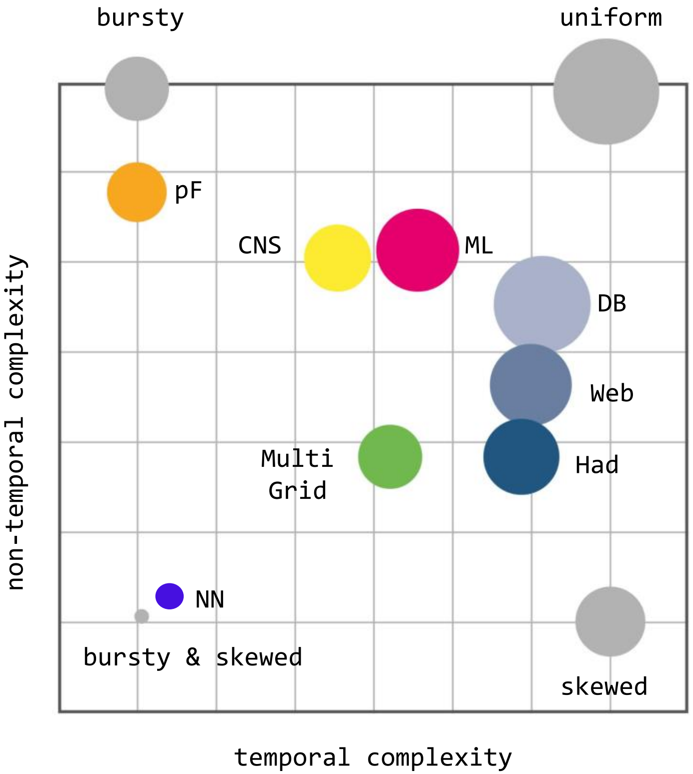


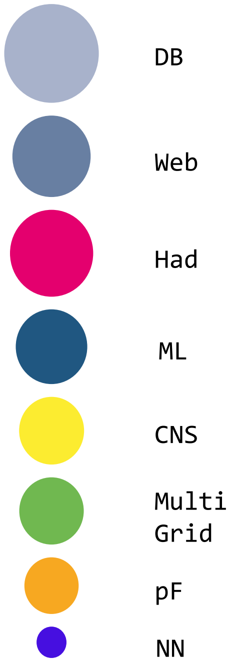
Griner et al., SIGMETRICS 2020

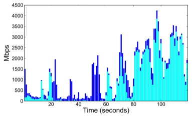
Recent Representation of Trace Structure:

Complexity Map







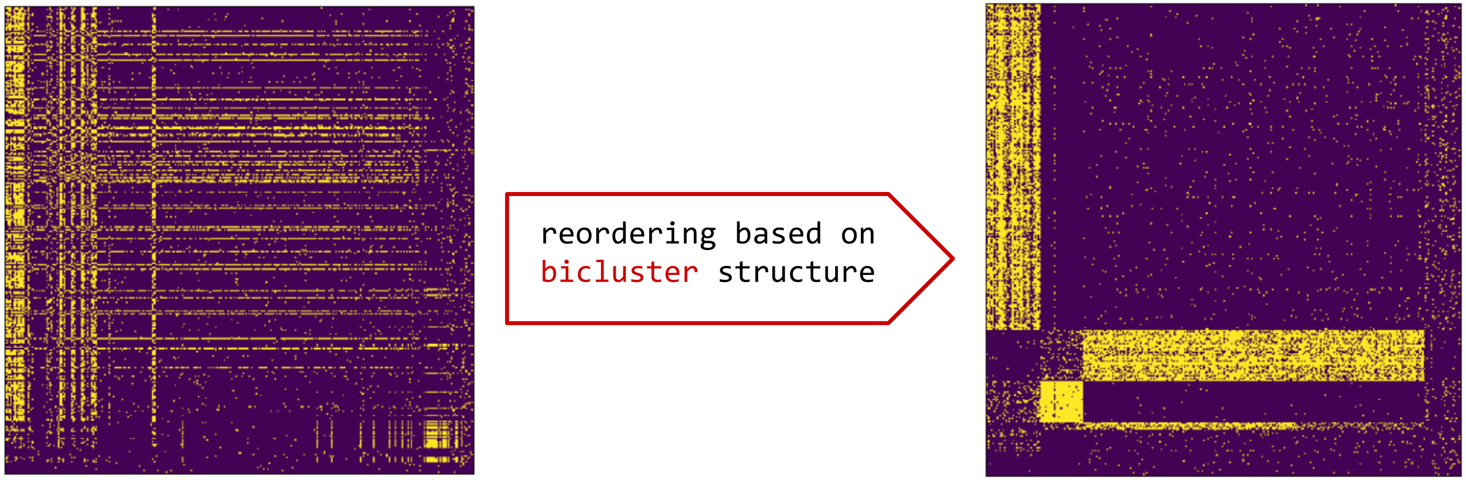


Different structures!

Griner et al., SIGMETRICS 2020

Traffic is also clustered:

Small Stable Clusters

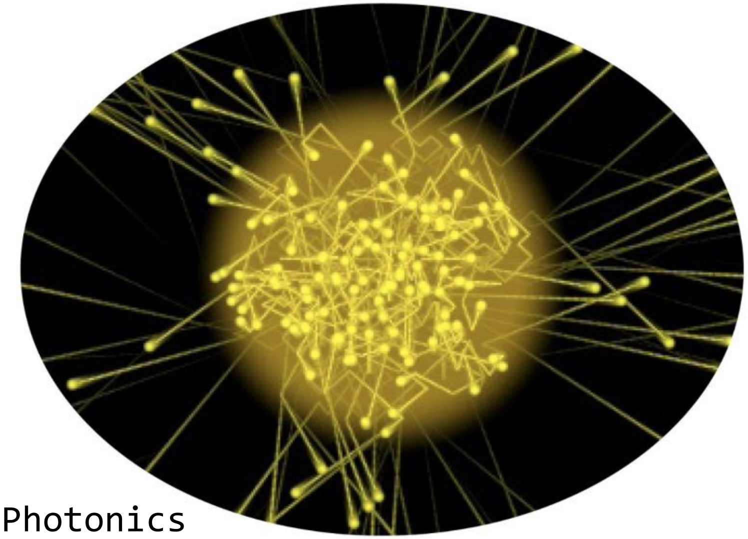


reordering based on bicluster structure

Opportunity: *exploit* with little reconfigurations!

Förster et al., Analyzing the Communication Clusters in Datacenters. WWW 2023

Sounds Crazy? Emerging Enabling Technology.



Photonics

H2020:

“Photonics one of only five key enabling technologies for future prosperity.”

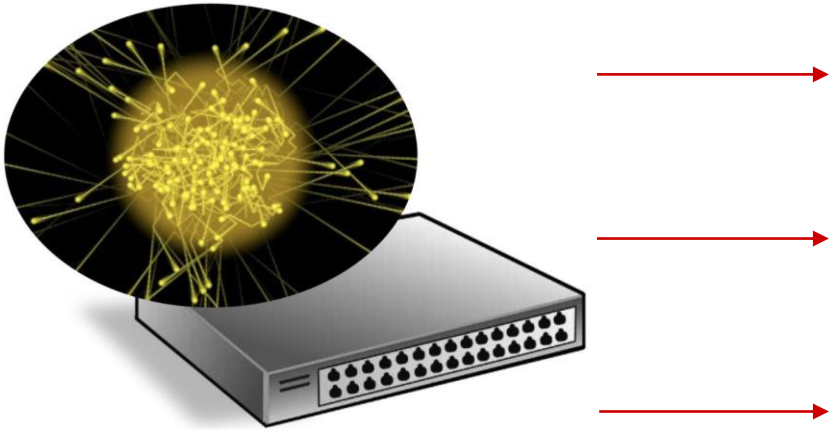
US National Research Council:

“Photons are the new Electrons.”

Enabler

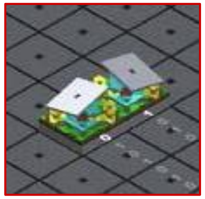
Novel Reconfigurable Optical Switches

- **Spectrum** of prototypes
  - Different sizes, different reconfiguration times
  - From our ACM **SIGCOMM** workshop OptSys



Prototype 1

Moving antenna (ms)



Prototype 2

Moving mirrors (mus)

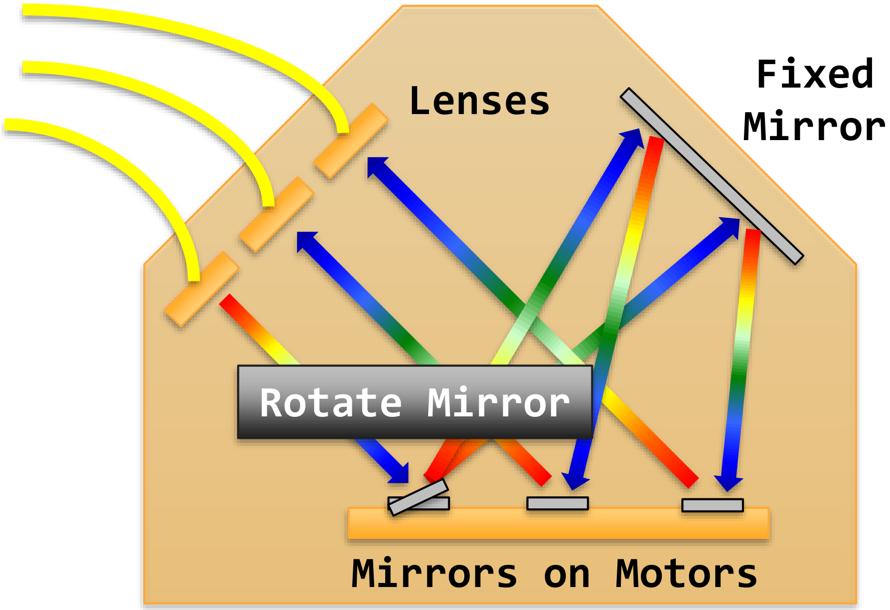
Prototype 3

Changing lambdas (ns)

Example

Optical Circuit Switch

- ⇢ Optical Circuit Switch rapid adaption of physical layer
  - ⇀ Based on rotating mirrors



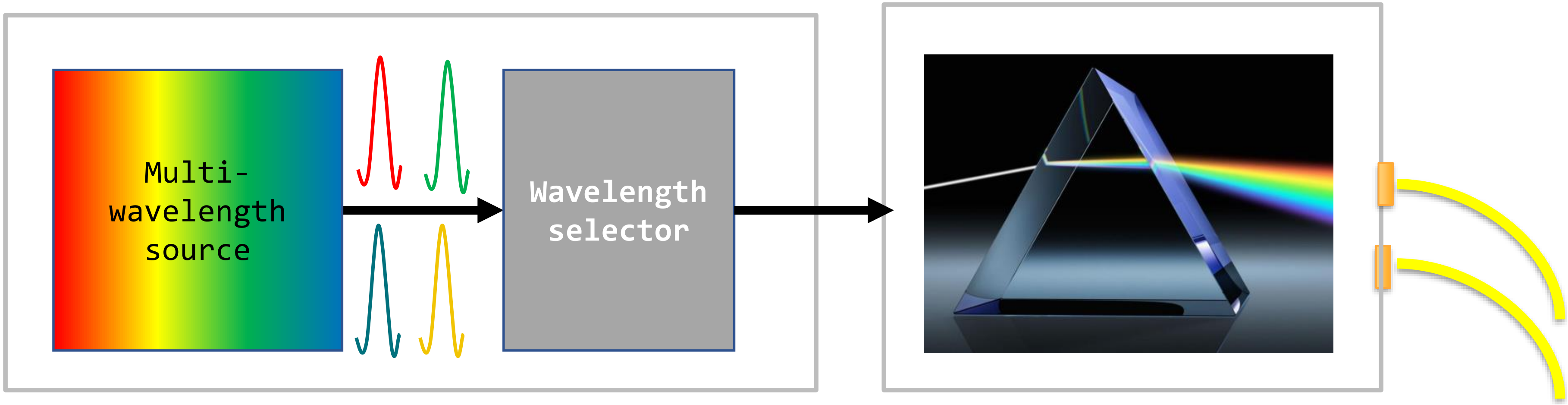
Optical Circuit Switch

By Nathan Farrington, SIGCOMM 2010

Another Example

Tunable Lasers

- ⇢ Depending on wavelength, forwarded differently
- ⇢ Optical switch is passive



Electrical switch with tunable laser

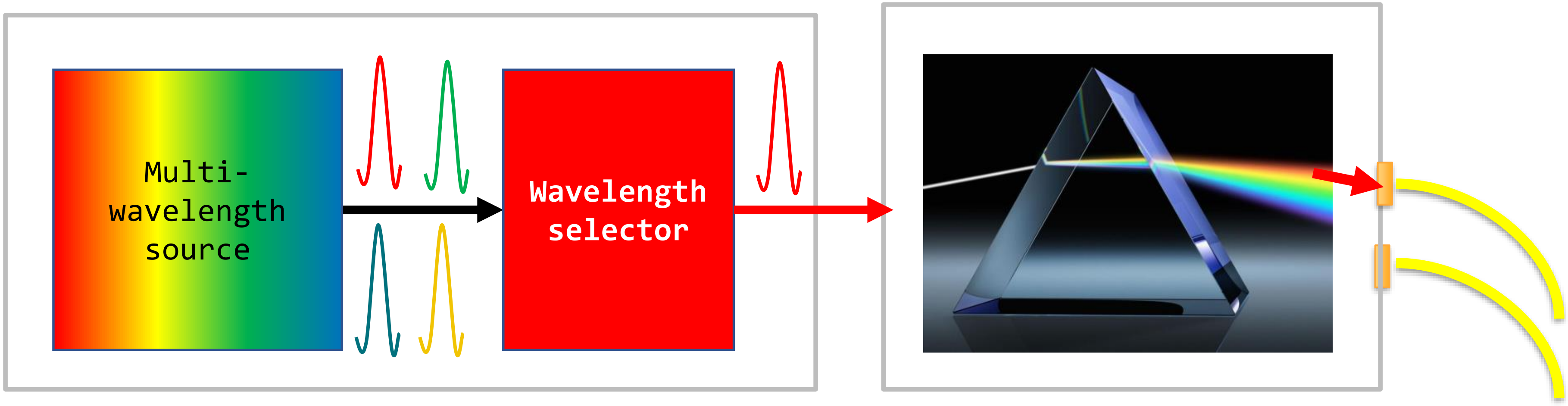
Optical switch *Passive*

Ballani et al., Sirius, ACM SIGCOMM 2020.

Another Example

Tunable Lasers

- Depending on wavelength, forwarded differently
- Optical switch is passive



Electrical switch with tunable laser

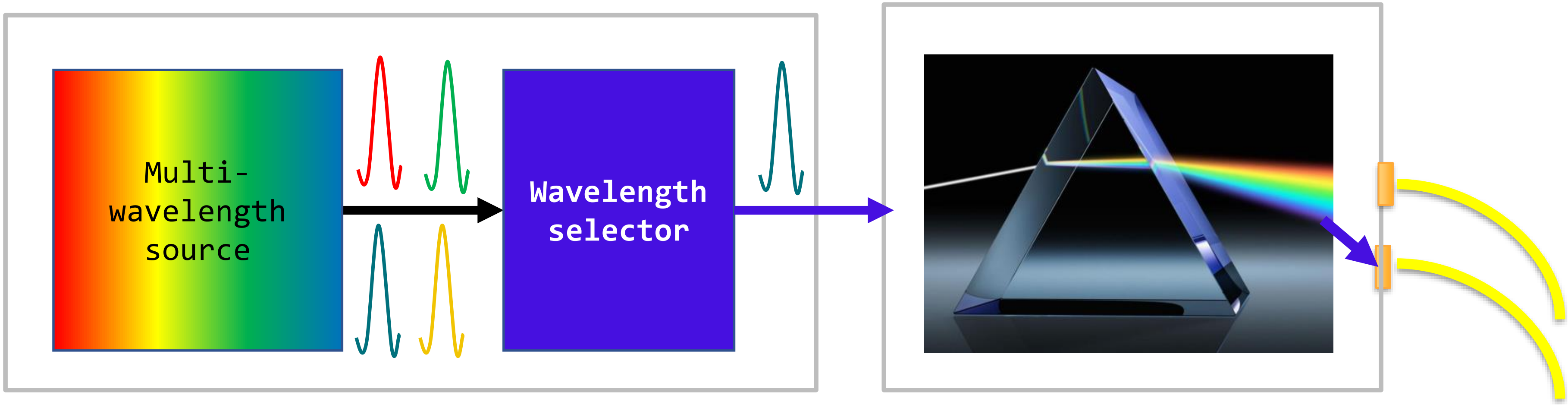
Optical switch
Passive

Ballani et al., Sirius, ACM SIGCOMM 2020.

Another Example

Tunable Lasers

- ⇢ Depending on wavelength, forwarded differently
- ⇢ Optical switch is passive



Electrical switch with tunable laser

Optical switch
Passive

Ballani et al., Sirius, ACM SIGCOMM 2020.

First Deployments

E.g., Google’s Datacenter Jupiter

Systems

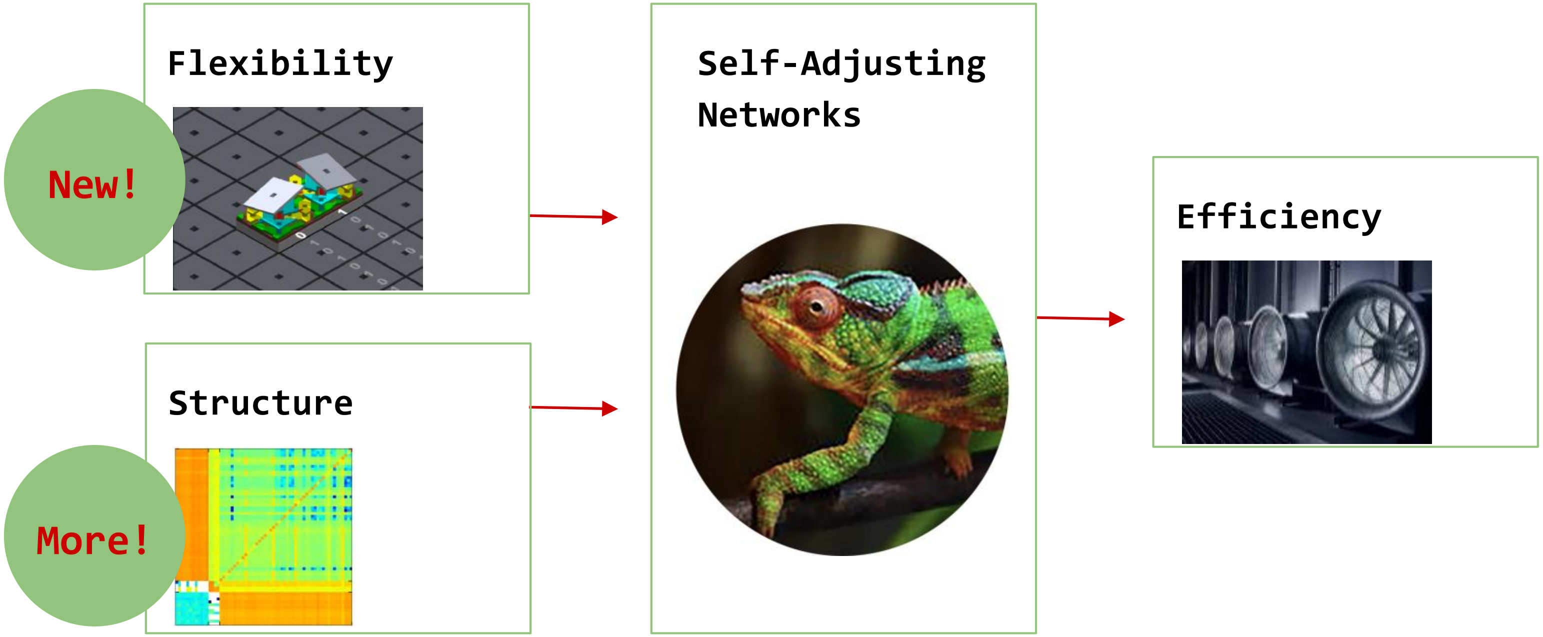
Jupiter evolving: Reflecting on Google’s data center network transformation

August 24, 2022

Amin Vahdat

VP & GM, Systems and Services Infrastructure

The Big Picture



New!

Flexibility

More!

Structure

Self-Adjusting Networks

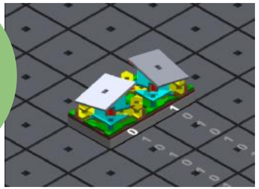
Efficiency

Now is the time!

The Big Picture

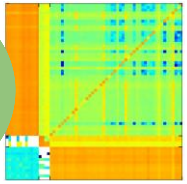
Flexibility

New!



Structure

More!



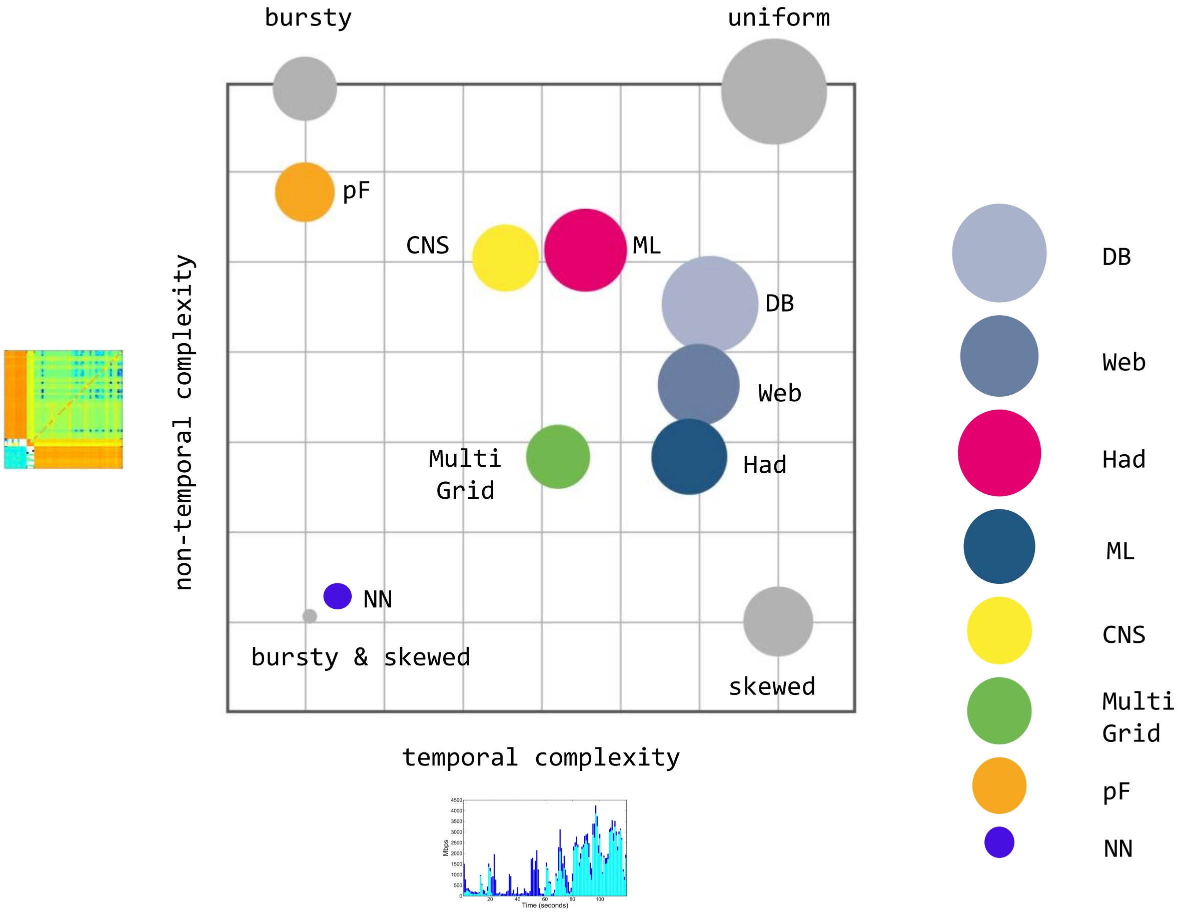
Self-Adjusting Networks

Efficiency

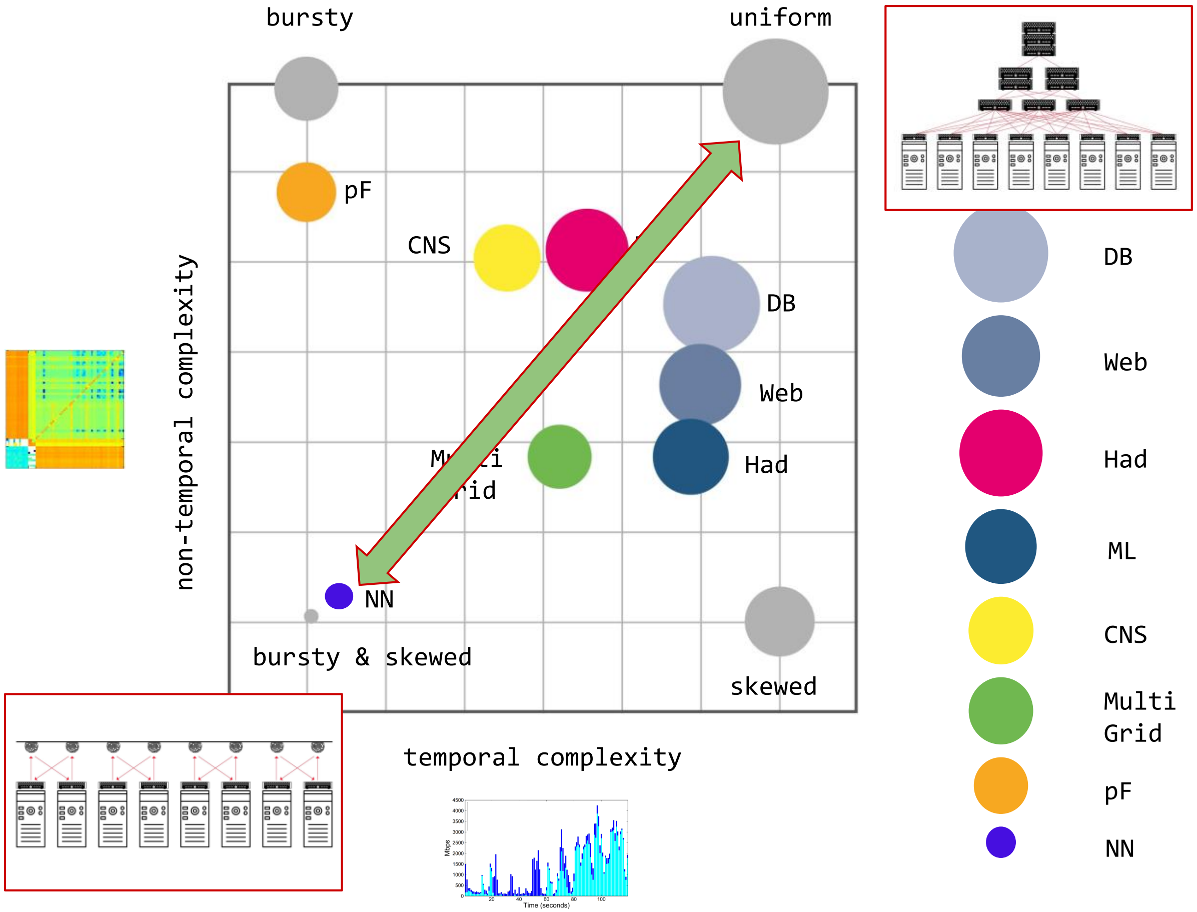
Now is the time!

Missing: Theoretical **foundations** of demand-aware, self-adjusting networks.

Potential Gain



Potential Gain



Unique Position

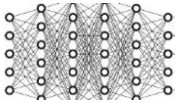
Demand-Aware, Self-Adjusting Systems

Everywhere, but mainly in software

Algorithmic trading

NETFLIX

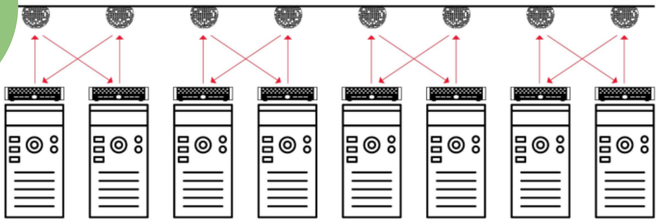
Recommender systems



Neural networks

vs

Our focus in this talk: in hardware



The Natural Question:

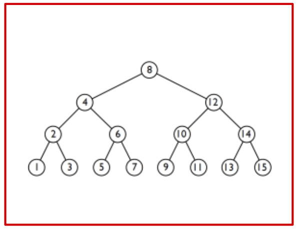
Given This Structure, What Can Be Achieved? Metrics and Algorithms?

A first insight: entropy of the demand.

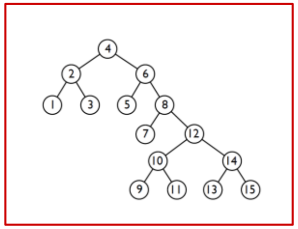
Insight:

Connection to Datastructures

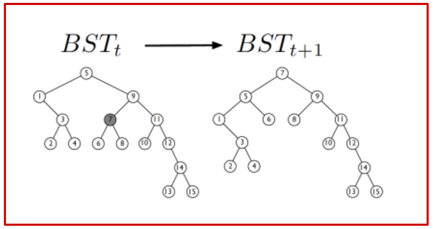
Traditional BST



Demand-aware BST



Self-adjusting BST

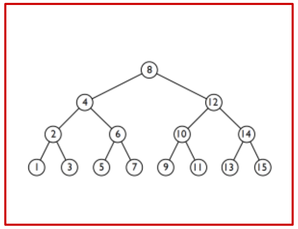


More structure: improved **access cost**

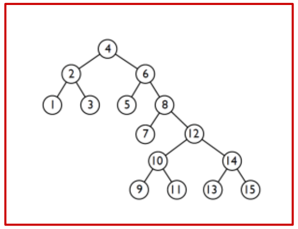
Insight:

Connection to Datastructures & Coding

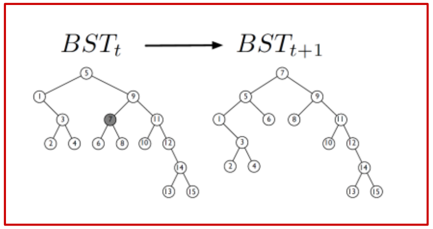
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)



Self-adjusting BST
(Dynamic Huffman coding)



More structure: improved **access cost** / shorter **codes**

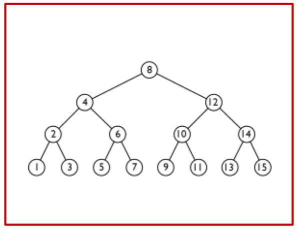
Insight:

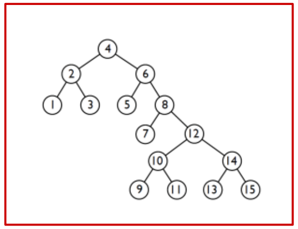
Connection to Datastructures & Coding

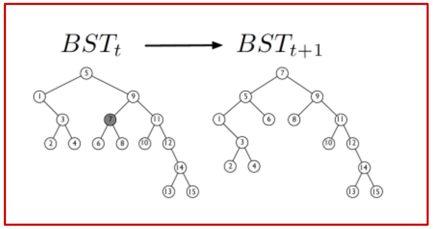
Traditional BST
(Worst-case coding)

Demand-aware BST
(Huffman coding)

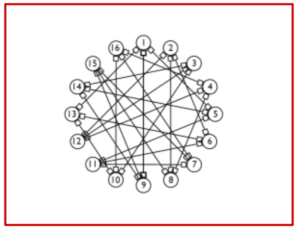
Self-adjusting BST
(Dynamic Huffman coding)

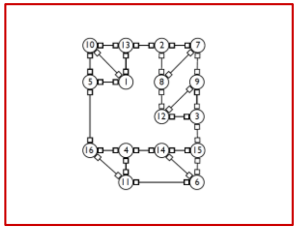


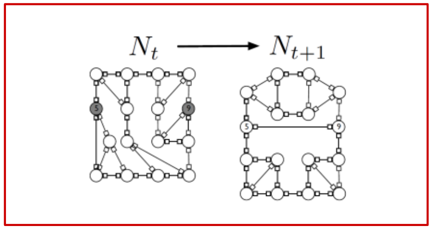




More structure: improved **access cost** / shorter **codes**





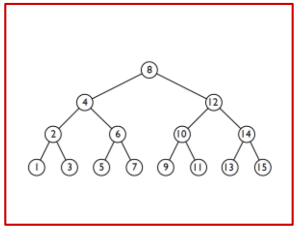


Similar **benefits**?

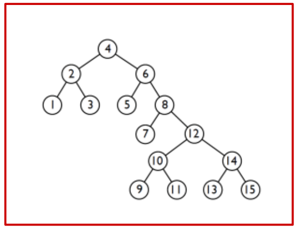
Insight:

Connection to Datastructures & Coding

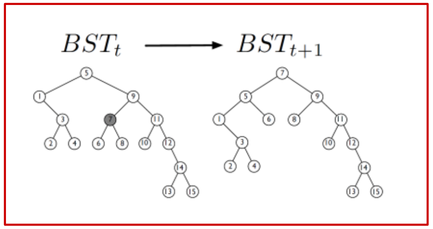
Traditional BST
(Worst-case coding)



Demand-aware BST
(Huffman coding)

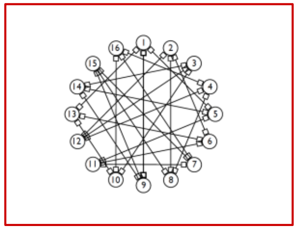


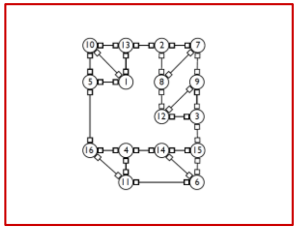
Self-adjusting BST
(Dynamic Huffman coding)

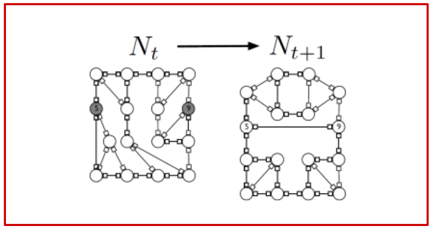


More than an analogy!

More structure: improved **access cost** / shorter **codes**







Similar **benefits**?

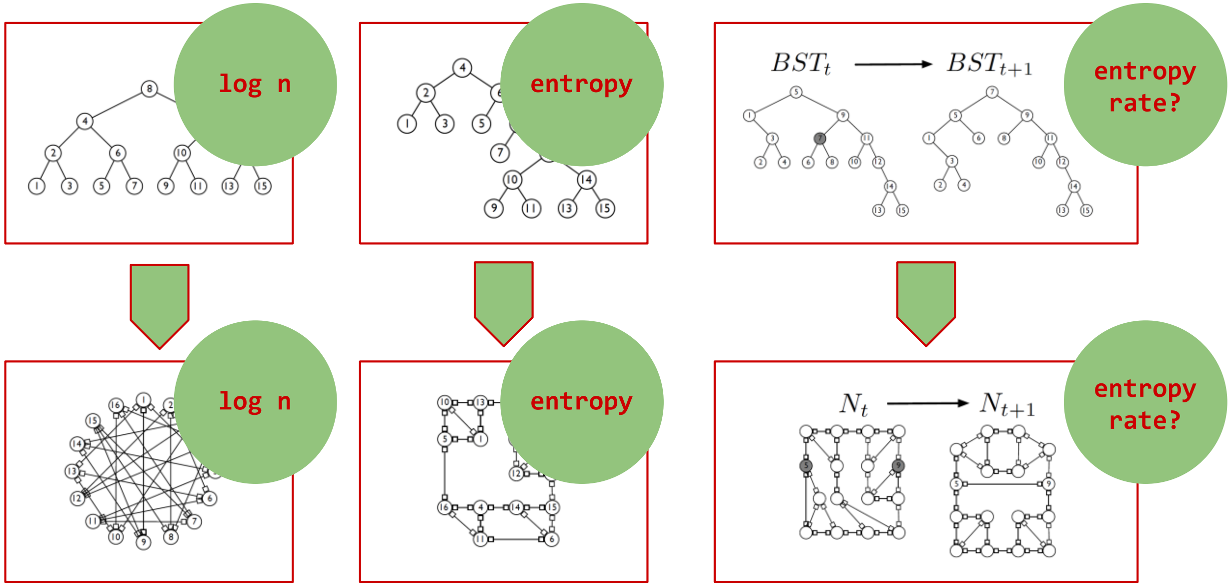
Insight:

Connection to Datastructures & Coding

Traditional BST
(Worst-case coding)

Demand-aware BST
(Huffman coding)

Self-adjusting BST
(Dynamic Huffman coding)



log n

entropy

entropy rate?

log n

entropy

entropy rate?

Reduced expected **route lengths**!

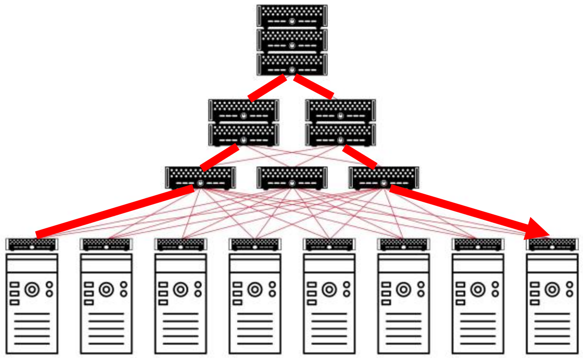
More than
an analogy!

Generalize methodology:
... and transfer entropy bounds and algorithms of data-structures to networks.

First result:
Demand-aware networks of asymptotically optimal route lengths.

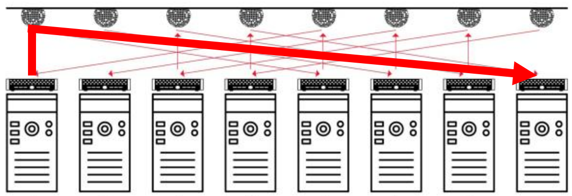
Reality more complicated

→ Self-adjusting networks may be really useful to serve large flows (elephant flows): avoiding multi-hop routing



6 hops

vs

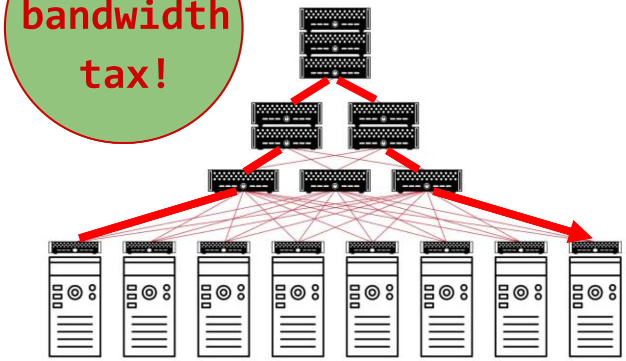


1 hop

Reality more complicated

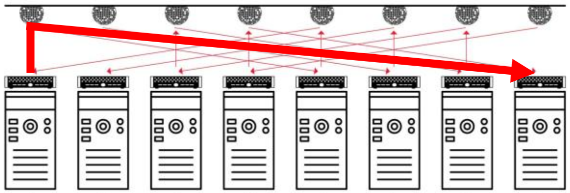
→ Self-adjusting networks may be really useful to serve large flows (elephant flows): avoiding multi-hop routing

bandwidth tax!



6 hops

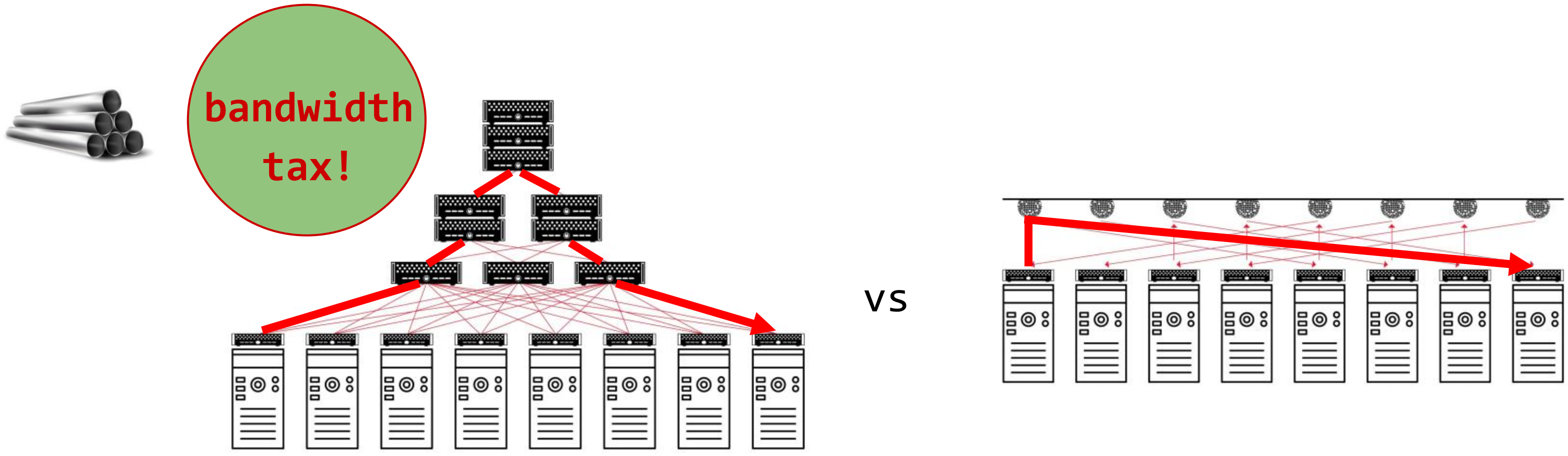
vs



1 hop

Reality more complicated

→ Self-adjusting networks may be really useful to serve large flows (elephant flows): avoiding multi-hop routing



bandwidth tax!

vs

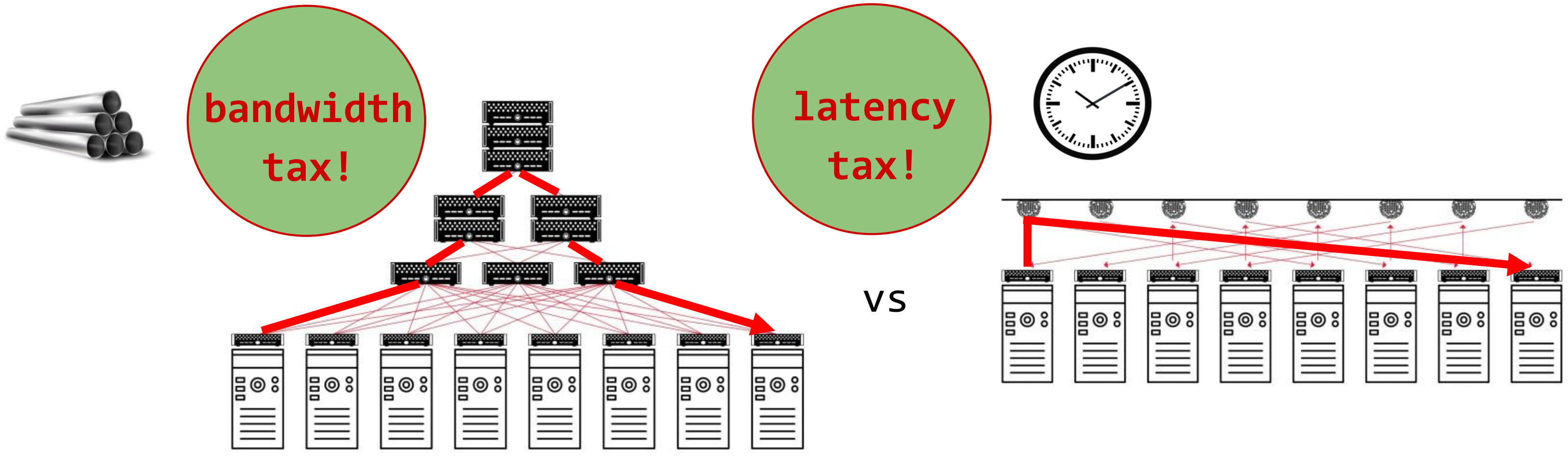
6 hops

1 hop

→ However, requires optimization and adaption, which takes time

Reality more complicated

→ Self-adjusting networks may be really useful to serve large flows (elephant flows): avoiding multi-hop routing



bandwidth tax!

latency tax!

vs

6 hops

1 hop

→ However, requires optimization and adaption, which takes time

Indeed, it is more complicated than that…

Challenge: Traffic Diversity

Diverse patterns:

⇀ Shuffling/Hadoop:
all-to-all

⇀ All-reduce/ML: ring or
tree traffic patterns

⇀ Elephant flows

⇀ Query traffic: skewed

⇀ Mice flows

⇀ Control traffic: does not evolve
but has non-temporal structure

Diverse requirements:

⇀ ML is bandwidth hungry,
small flows are latency-sensitive

Shuffling
All-to-All

ML
Large flows

Delay sensitive

Telemetry / control

Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and
demand-aware

Demand-oblivious

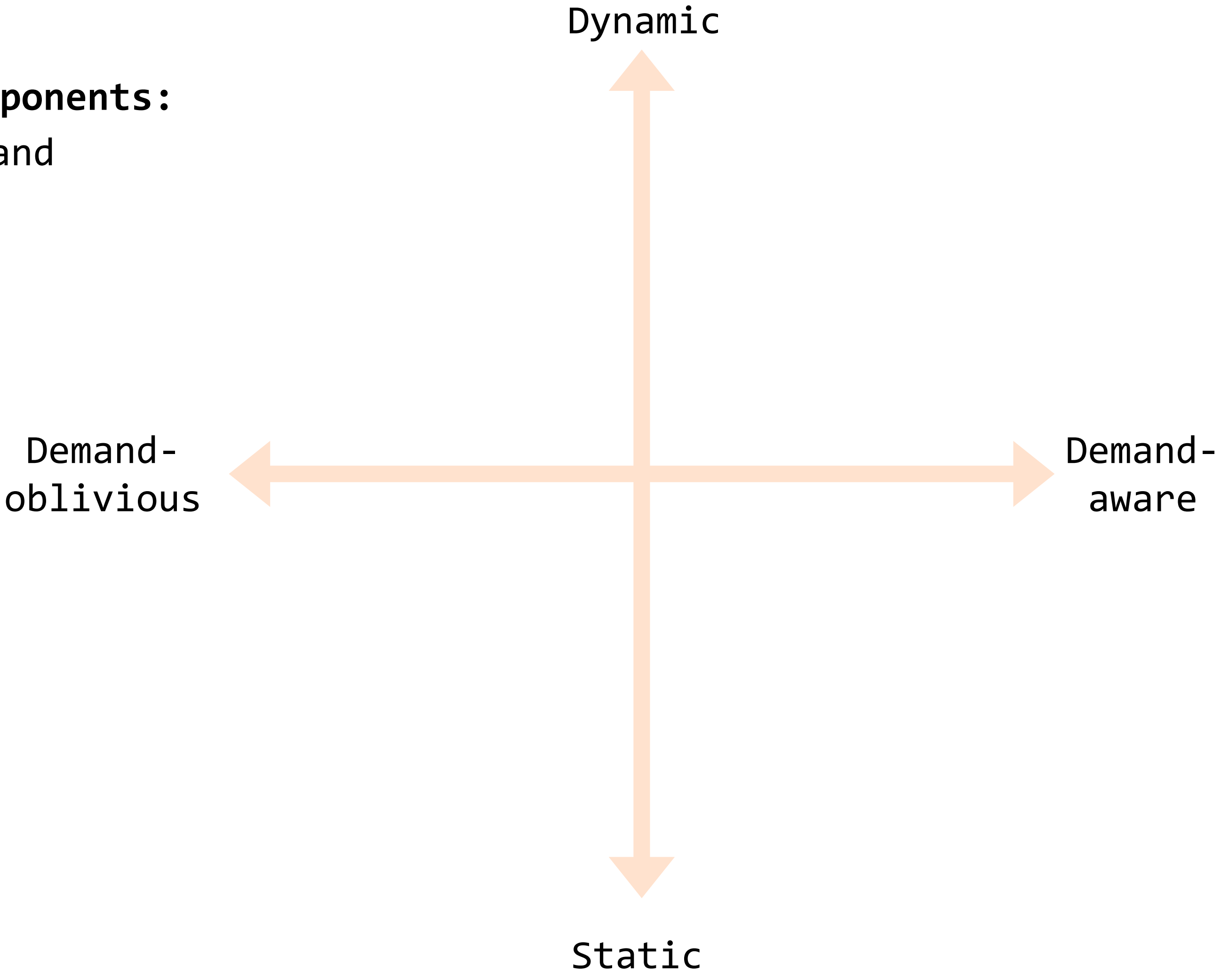
Demand-aware

Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic

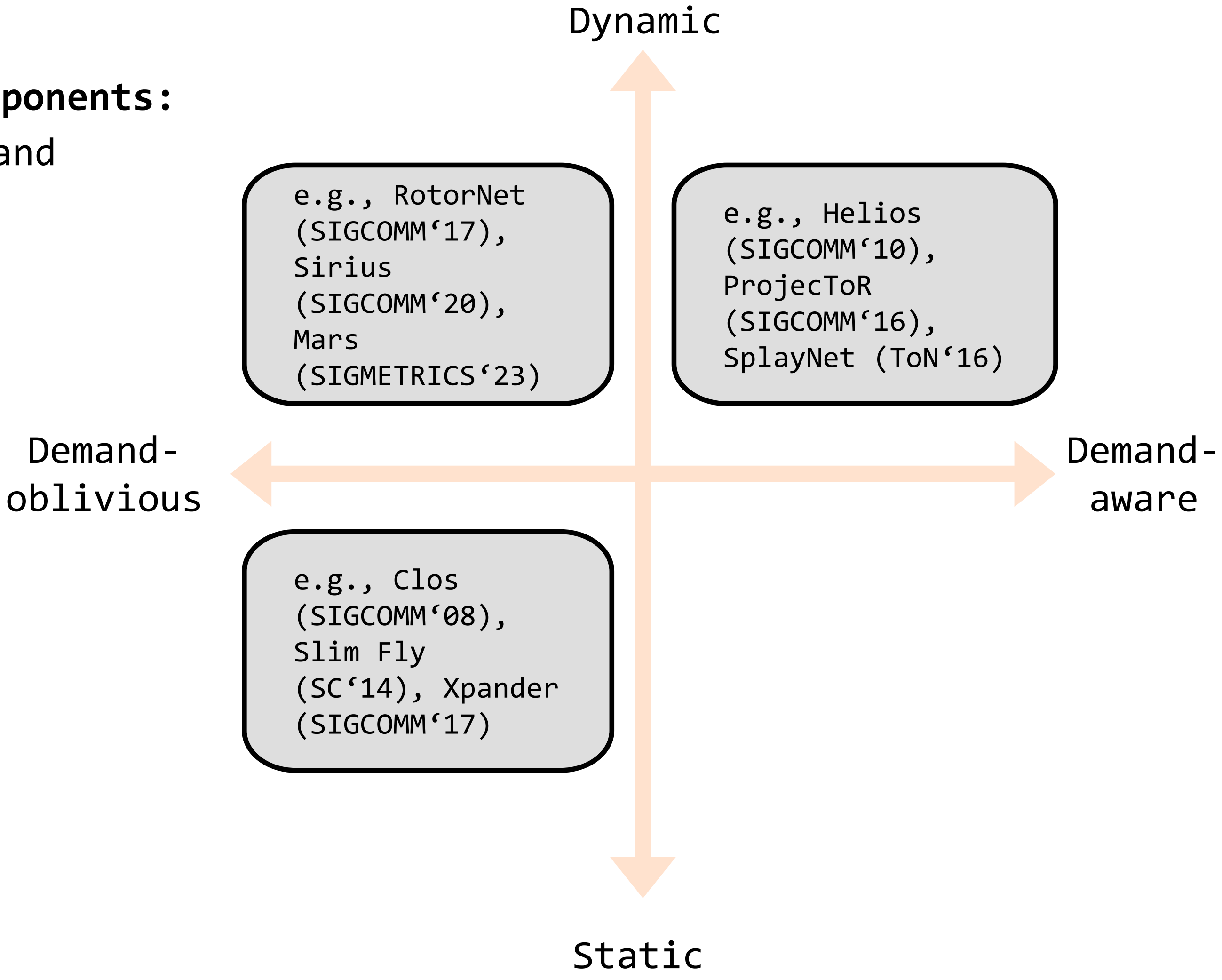


Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic



Dynamic

e.g., RotorNet (SIGCOMM‘17), Sirius (SIGCOMM‘20), Mars (SIGMETRICS‘23)

e.g., Helios (SIGCOMM‘10), ProjecToR (SIGCOMM‘16), SplayNet (ToN‘16)

Demand-oblivious

Demand-aware

e.g., Clos (SIGCOMM‘08), Slim Fly (SC‘14), Xpander (SIGCOMM‘17)

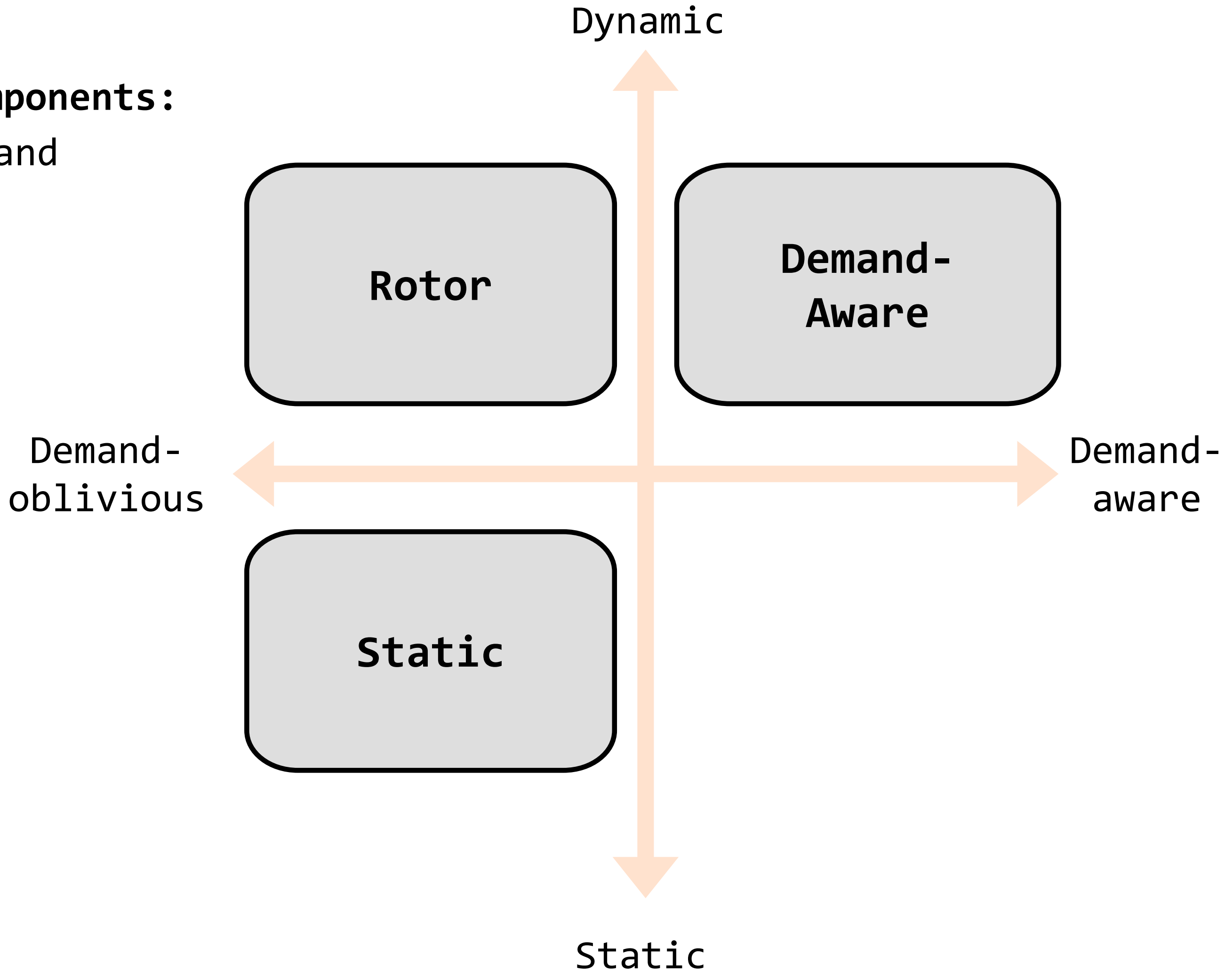
Static

Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic

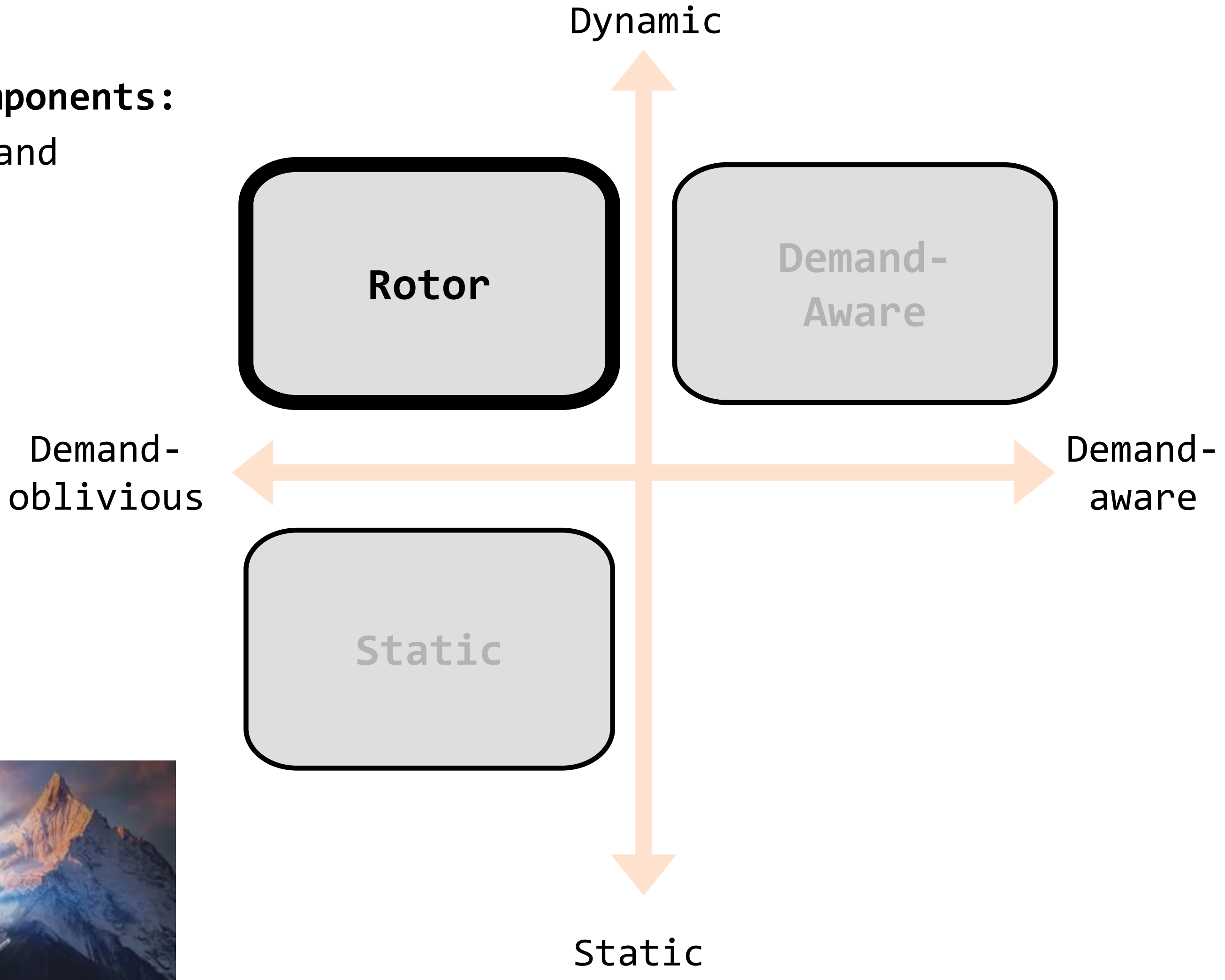


Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic



Dynamic

Rotor

Demand-Aware

Demand-oblivious

Demand-aware

Static

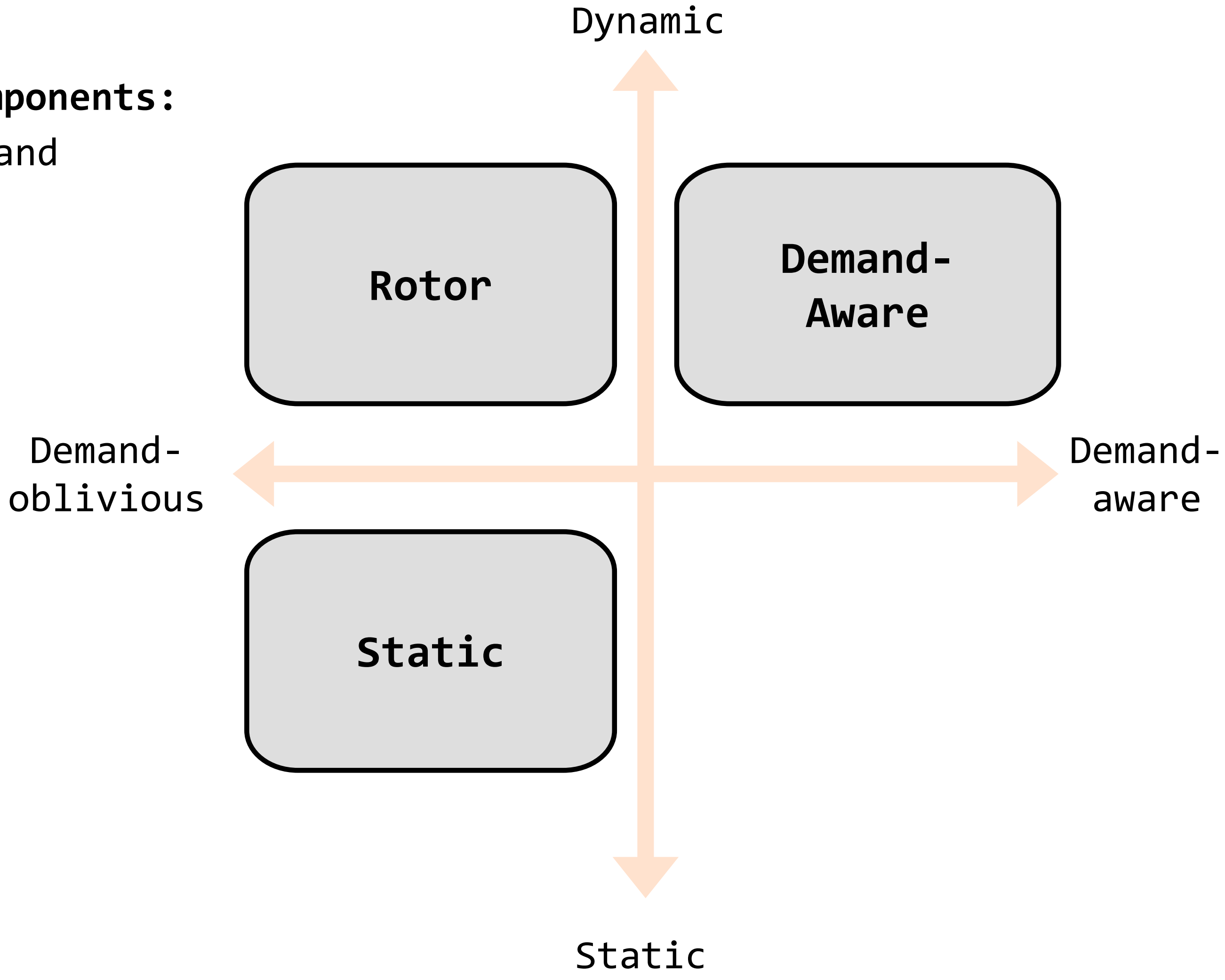
Static

Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic

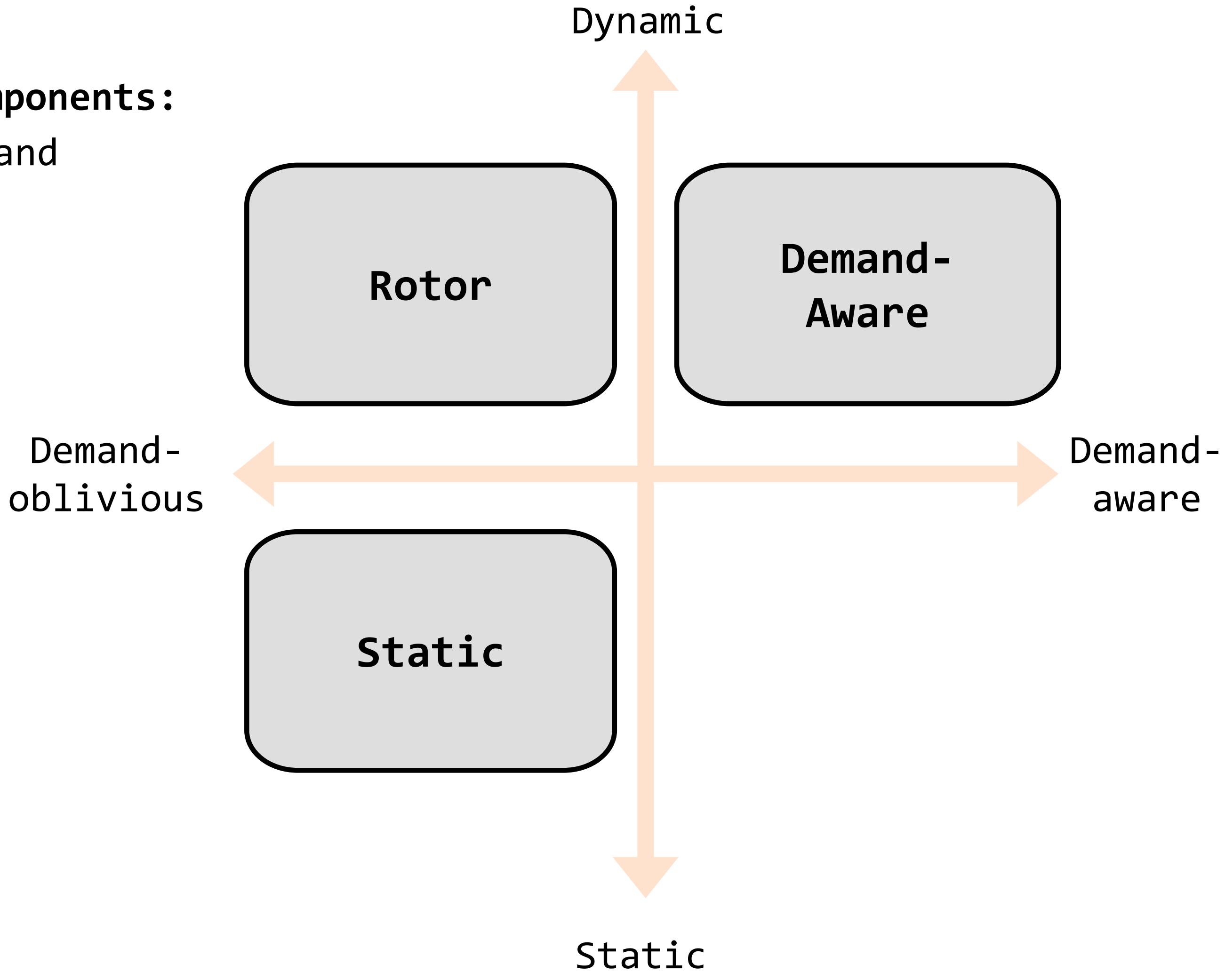


Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic



Dynamic

Rotor

Demand-Aware

Demand-oblivious

Demand-aware

Static

Static

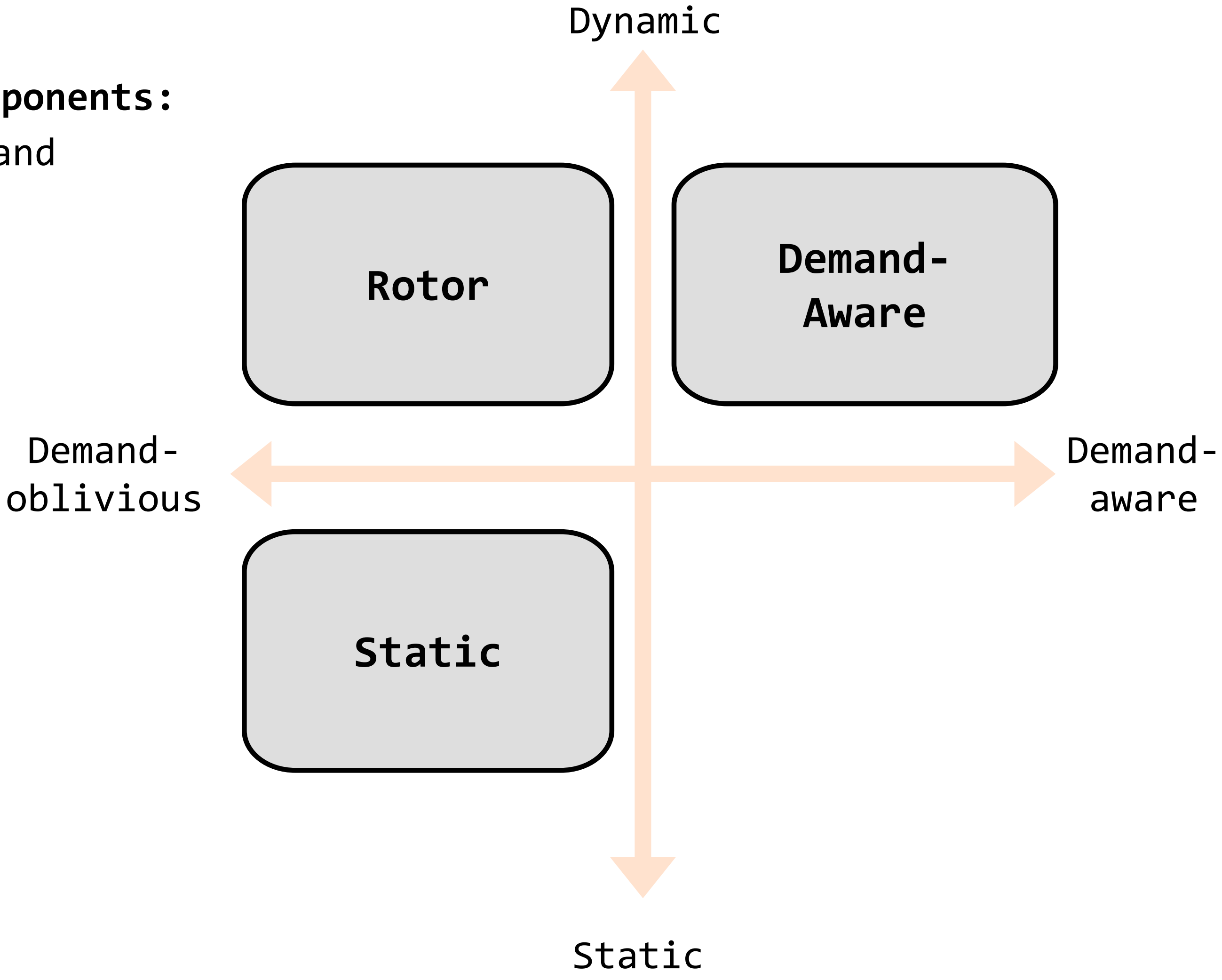
Which approach is best?

Opportunity: Tech Diversity

Diverse topology components:

⇀ demand-oblivious and demand-aware

⇀ static vs dynamic



Dynamic

Rotor

Demand-Aware

Demand-oblivious

Demand-aware

Static

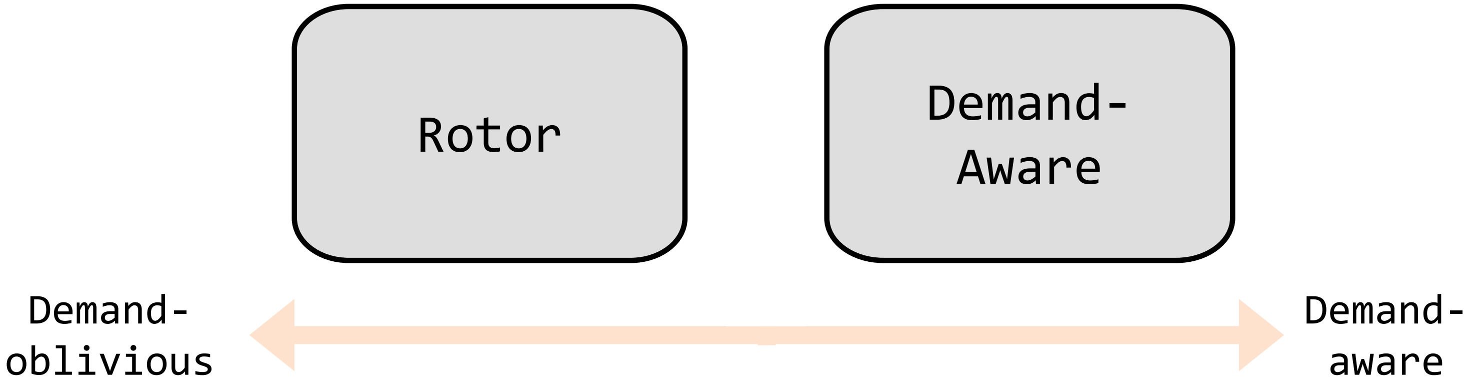
Static

Which approach is best?

**As always in CS:
It depends…**

Design Tradeoffs (1)

The “Awareness-Dimension”



Good for all-to-all traffic!

→ oblivious: very fast periodic direct connectivity

→ no control plane overhead

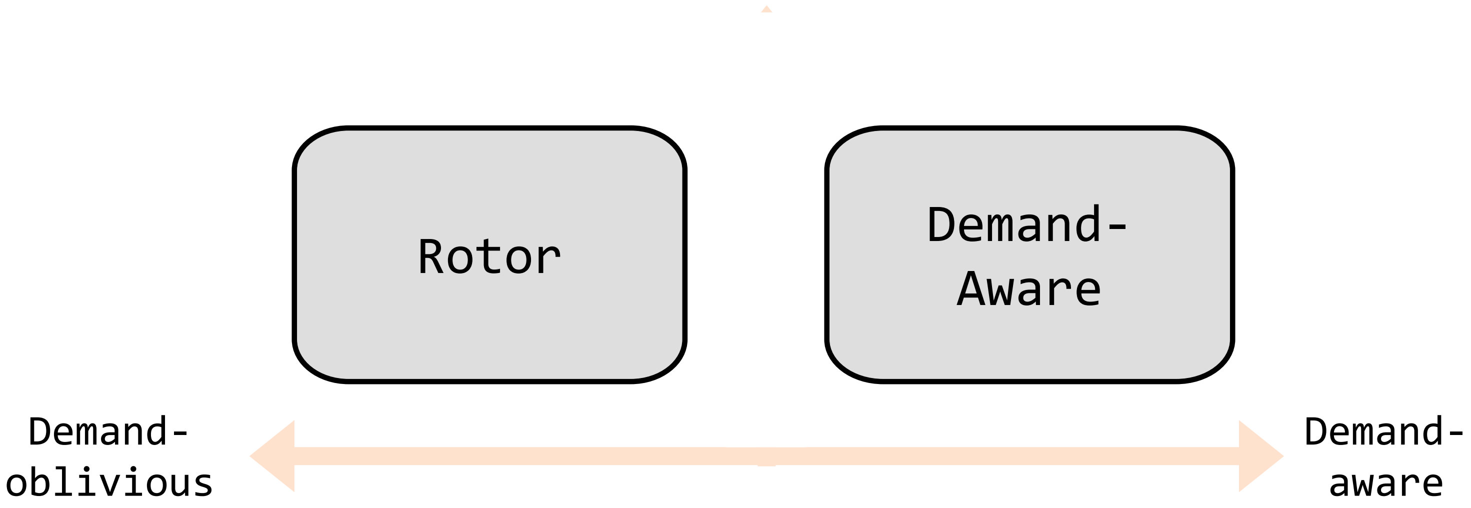
Good for elephant flows!

→ optimizable toward traffic

→ but slower

Design Tradeoffs (1)

The “Awareness-Dimension”



Rotor

Demand-Aware

Demand-oblivious

Demand-aware

Good for all-to-all traffic!

→ oblivious: very fast periodic direct connectivity

→ no control plane overhead

Good for elephant flows!

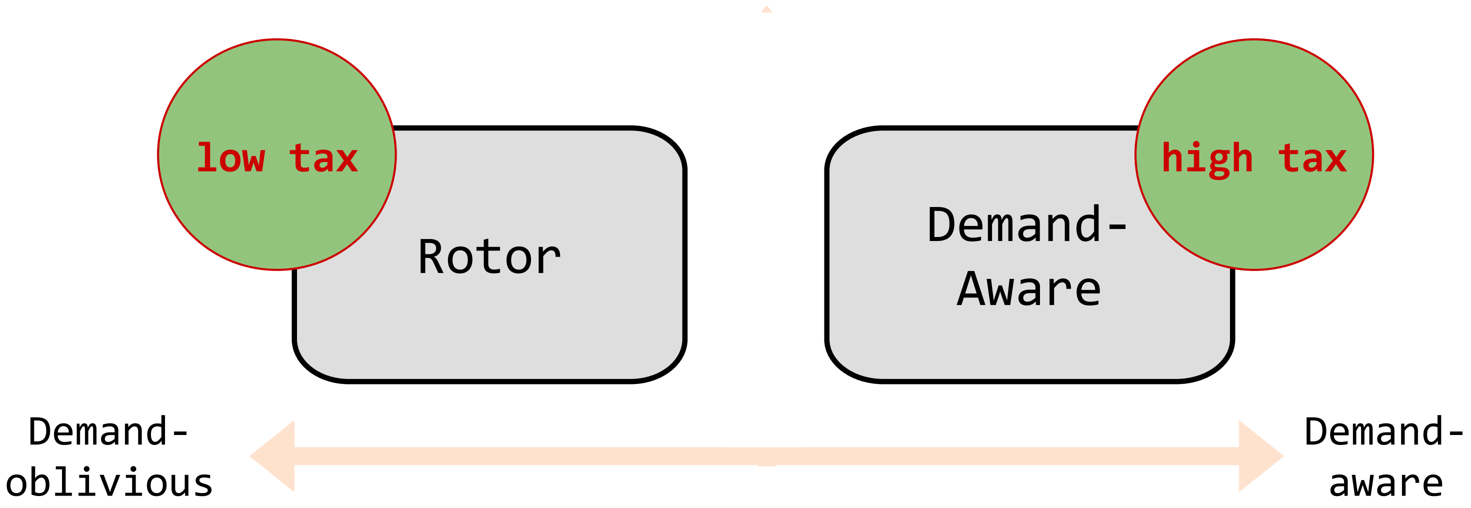
→ optimizable toward traffic

→ but slower

Compared to static networks: latency tax!

Design Tradeoffs (1)

The “Awareness-Dimension”



low tax

Rotor

Demand-Aware

high tax

Demand-oblivious

Demand-aware

Good for all-to-all traffic!

→ oblivious: very fast periodic direct connectivity

→ no control plane overhead

Good for elephant flows!

→ optimizable toward traffic

→ slower: requires optimization, collecting data, …

Compared to static networks: latency tax!

Design Tradeoffs (2)

The “Flexibility-Dimension”

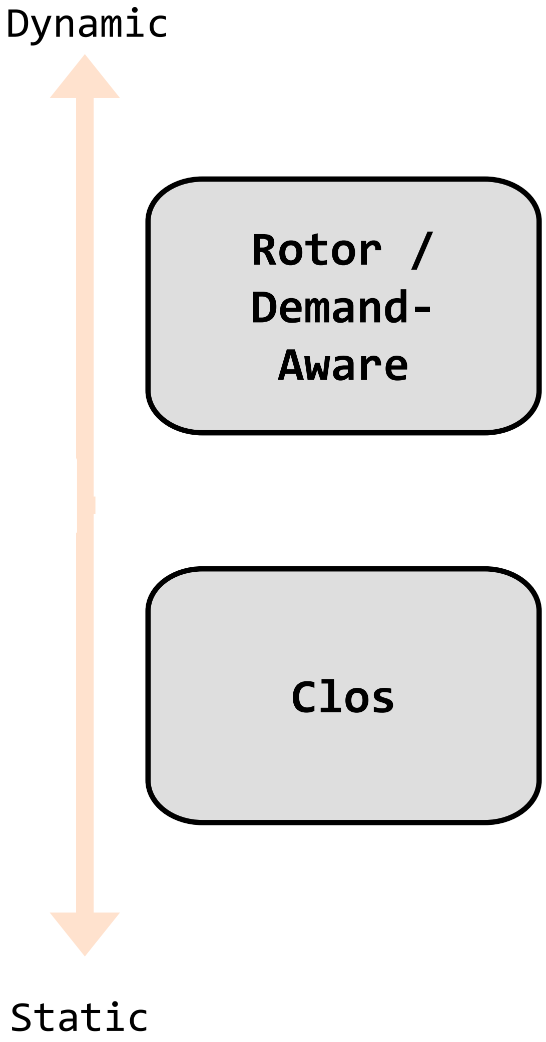
Good for high throughput!

⇀ direct connectivity saves bandwidth along links

Good for low latency!

⇀ no need to wait for reconfigurable links

⇀ **compared to dynamic: bandwidth tax (multi-hop)**



Dynamic

Rotor / Demand-Aware

Clos

Static

Design Tradeoffs (2)

The “Flexibility-Dimension”

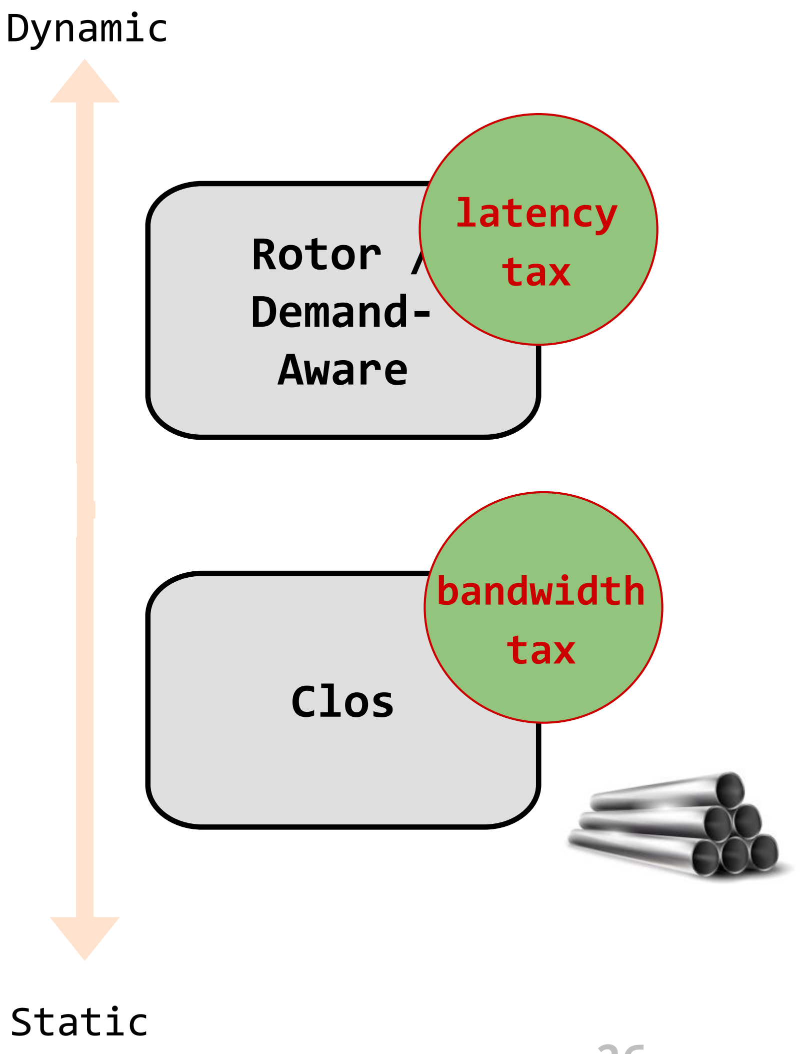
Good for high throughput!

⇀ direct connectivity saves bandwidth along links

Good for low latency!

⇀ no need to wait for reconfigurable links

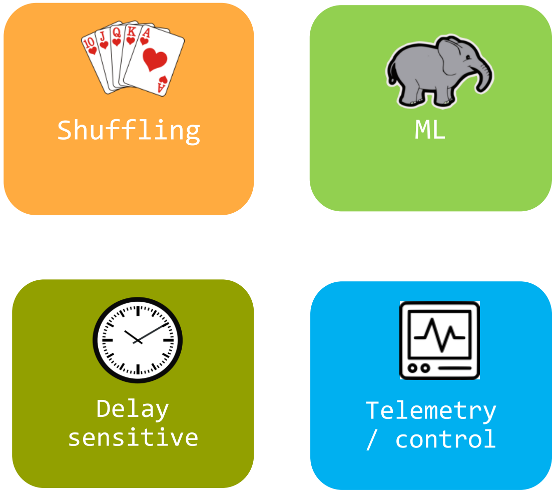
⇀ **compared to dynamic: bandwidth tax (multi-hop)**



First Observations

- **Observation 1:** Different topologies provide different tradeoffs.
- **Observation 2:** Different traffic requires different topology types.
- **Observation 3:** A **mismatch of demand** and topology can increase **flow completion times**.

Examples:
Match or Mismatch?



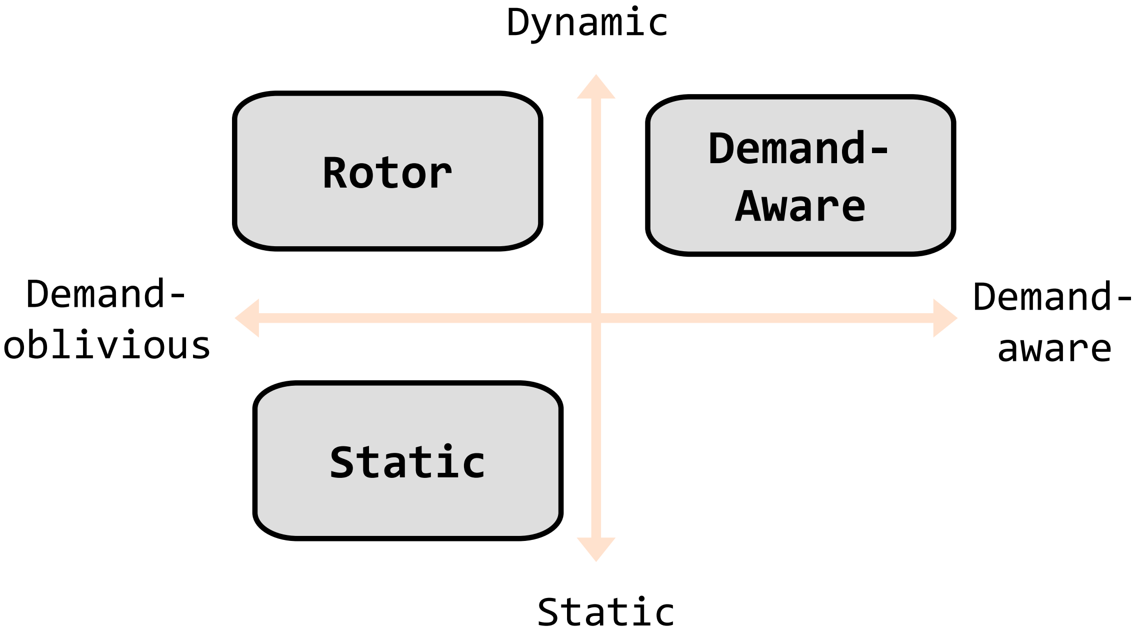
Shuffling

ML

Delay sensitive

Telemetry / control

Demand



Dynamic

Rotor

Demand-Aware

Demand-oblivious

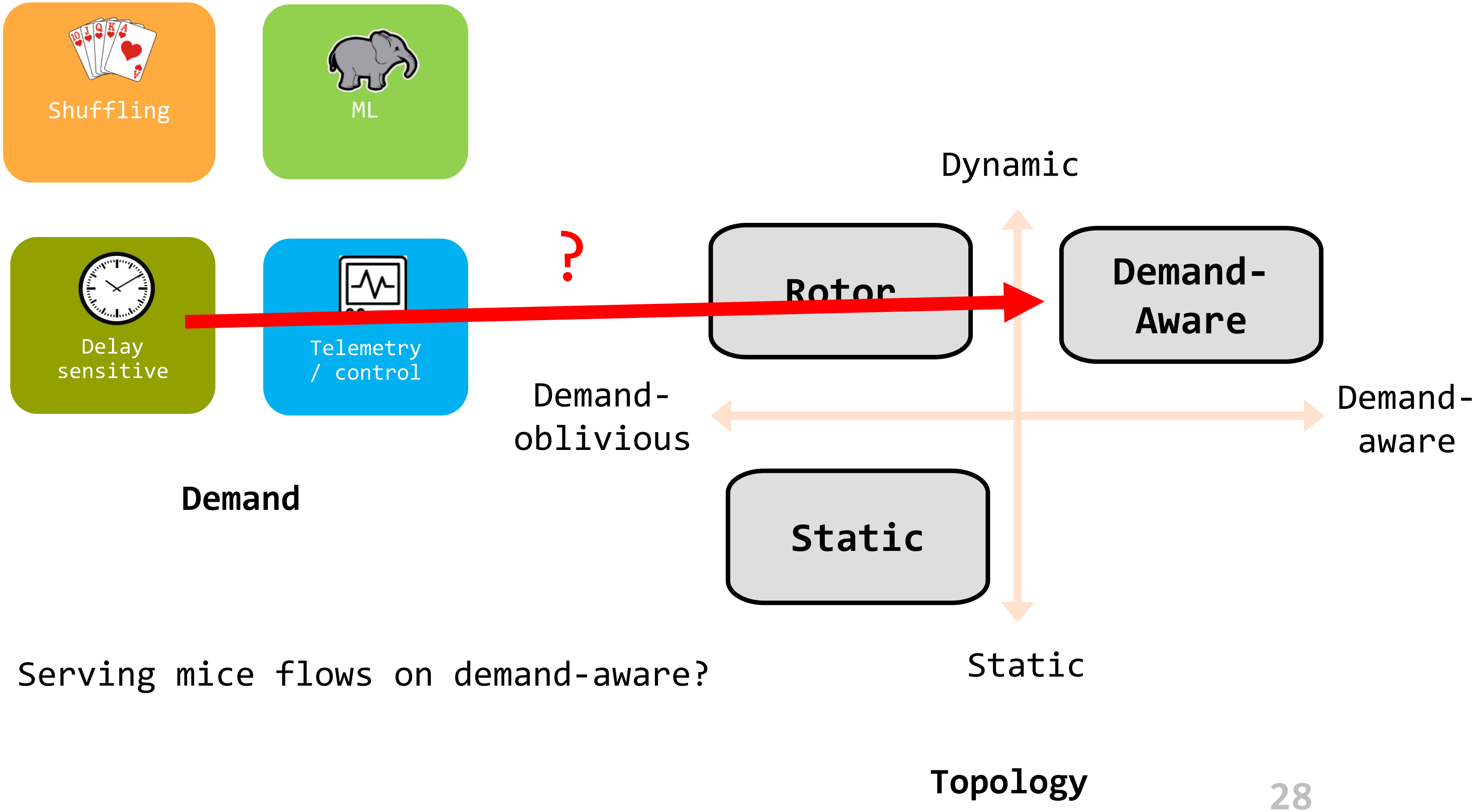
Demand-aware

Static

Static

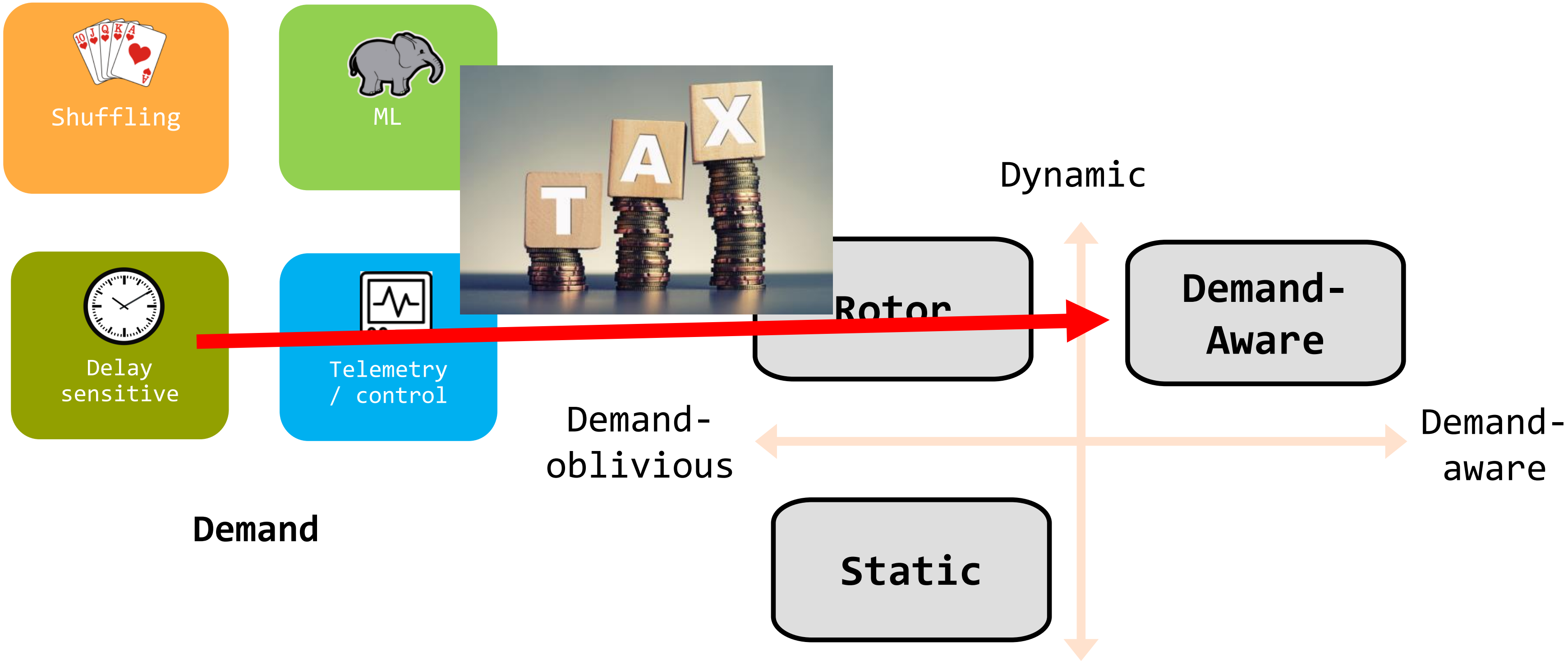
Topology

Examples:
Match or Mismatch?



Serving mice flows on demand-aware?

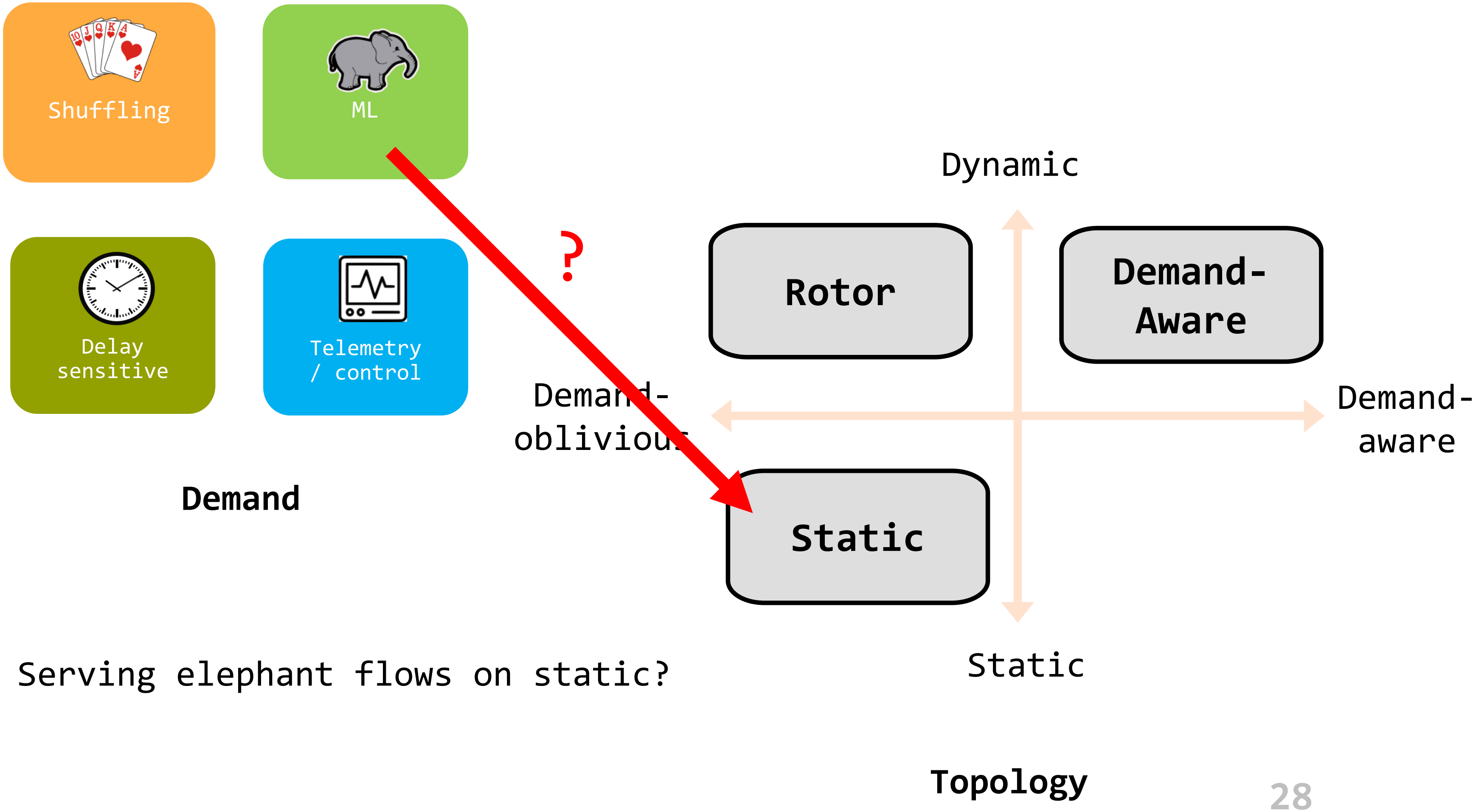
Examples:
Match or Mismatch?



Serving mice flows on demand-aware?

Bad idea! Latency tax.

Examples:
Match or Mismatch?



Shuffling

ML

Delay sensitive

Telemetry / control

Demand

?

Dynamic

Rotor

Demand-Aware

Demand-oblivious

Demand-aware

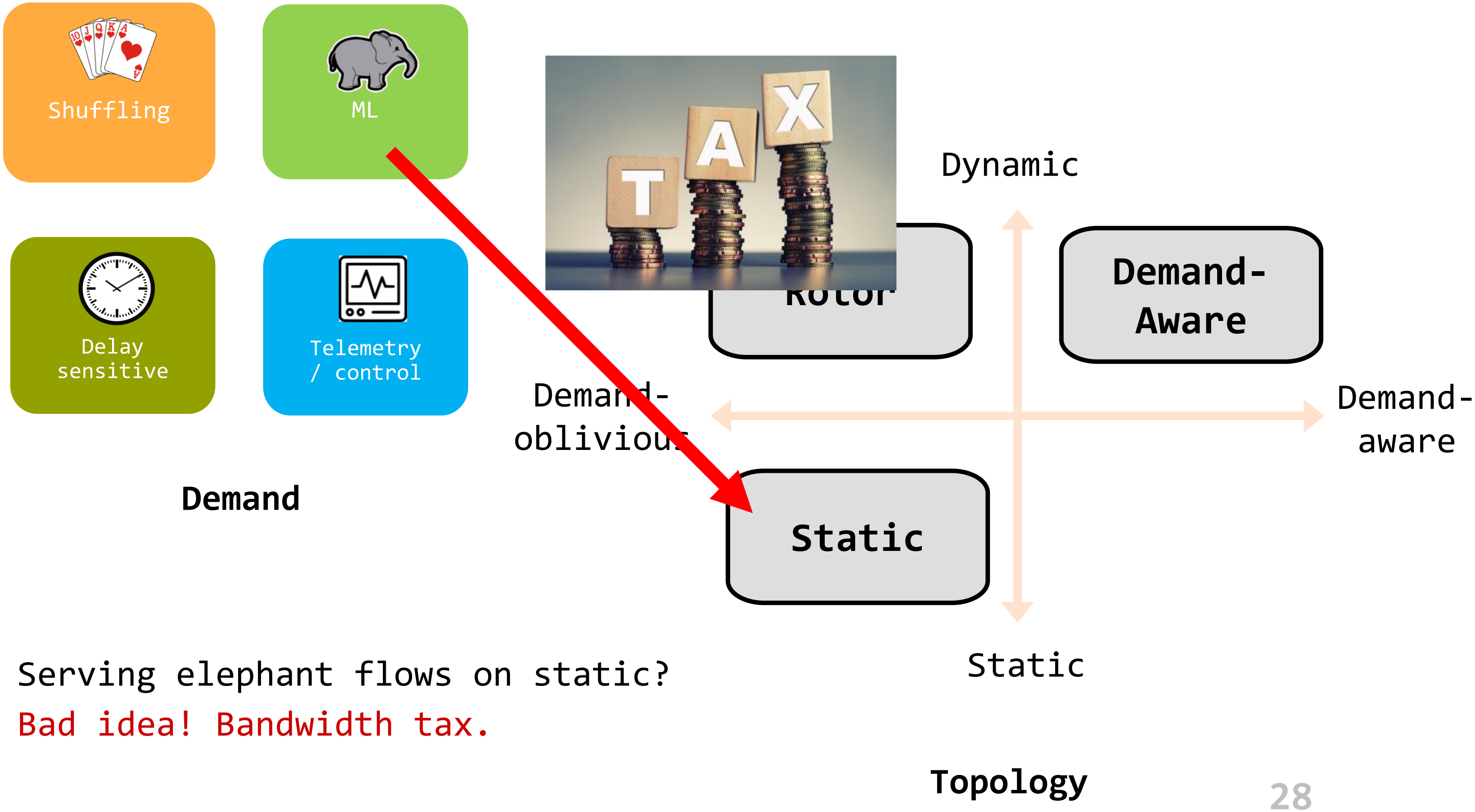
Static

Static

Topology

Serving elephant flows on static?

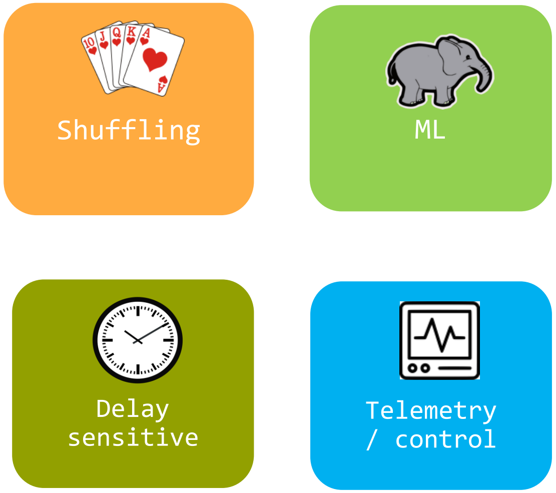
Examples:
Match or Mismatch?



Serving elephant flows on static?

Bad idea! Bandwidth tax.

Examples:
Match or Mismatch?



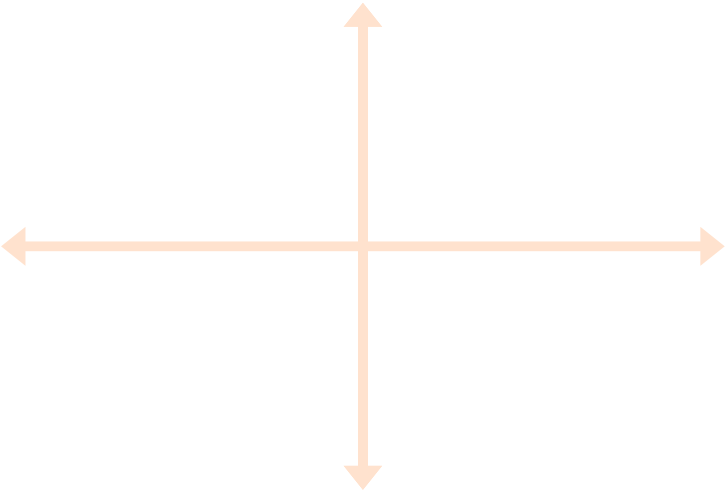
Shuffling

ML

Delay sensitive

Telemetry / control

Demand



Dynamic

Demand-oblivious

Demand-aware

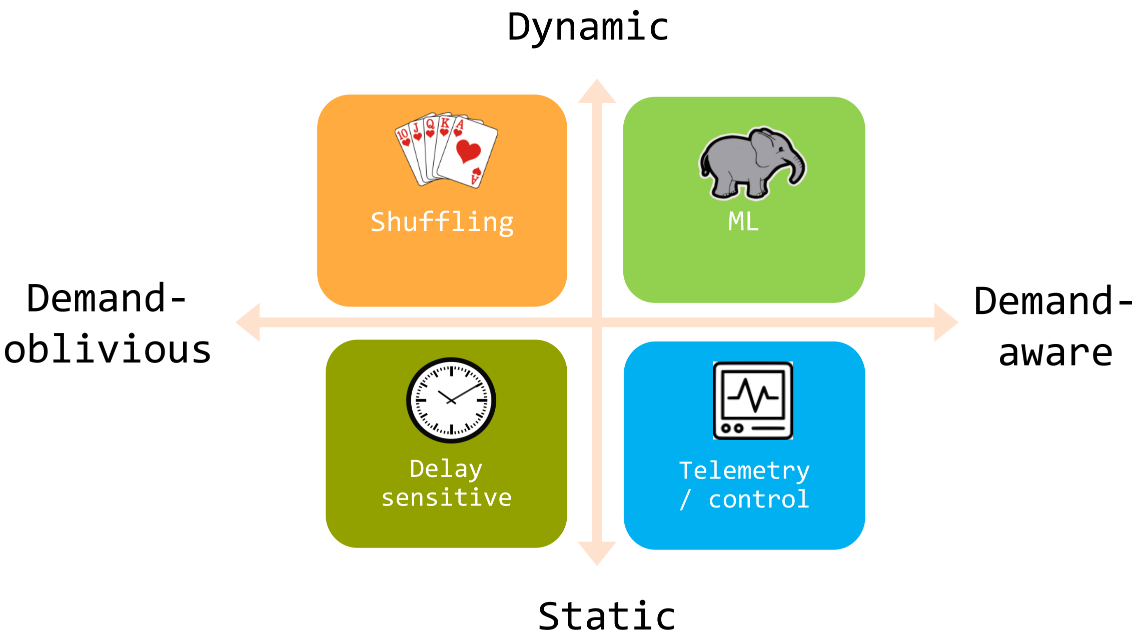
Static

Topology

Serving elephant flows on static?

Bad idea! Bandwidth tax.

A Solution: Cerberus



We have a first approach:

Cerberus** serves traffic on the “best topology”! (Optimality open)

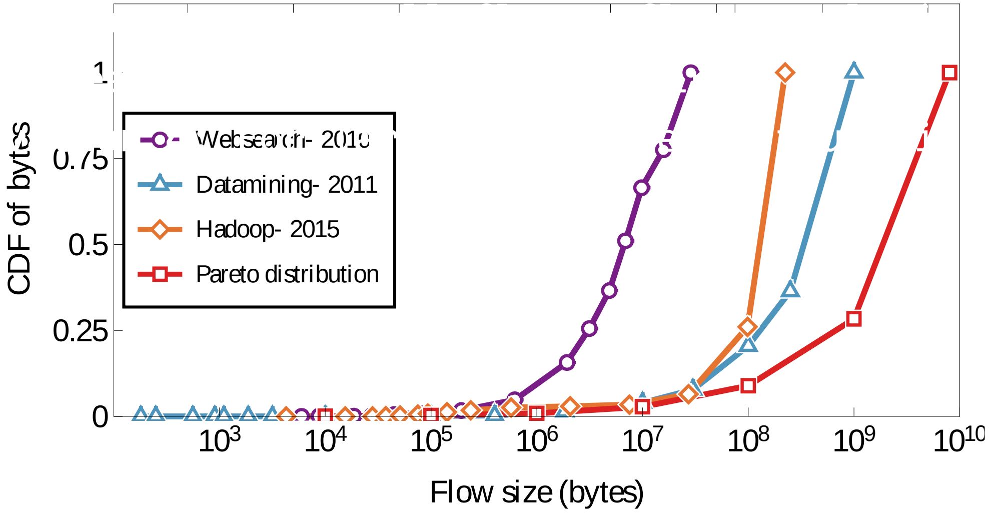
* Griner et al., ACM SIGMETRICS 2022

Flow Size Matters

On what should topology type depend? We argue: flow size.

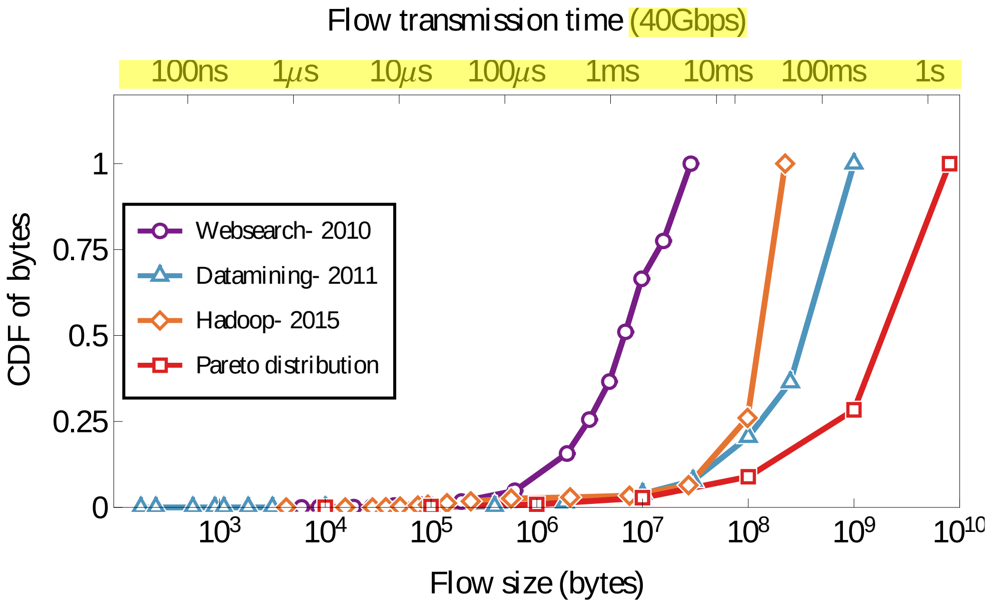
Flow Size Matters

On what should topology type depend? We argue: flow size.



⇢ **Observation 1:** Different apps have different flow size distributions.

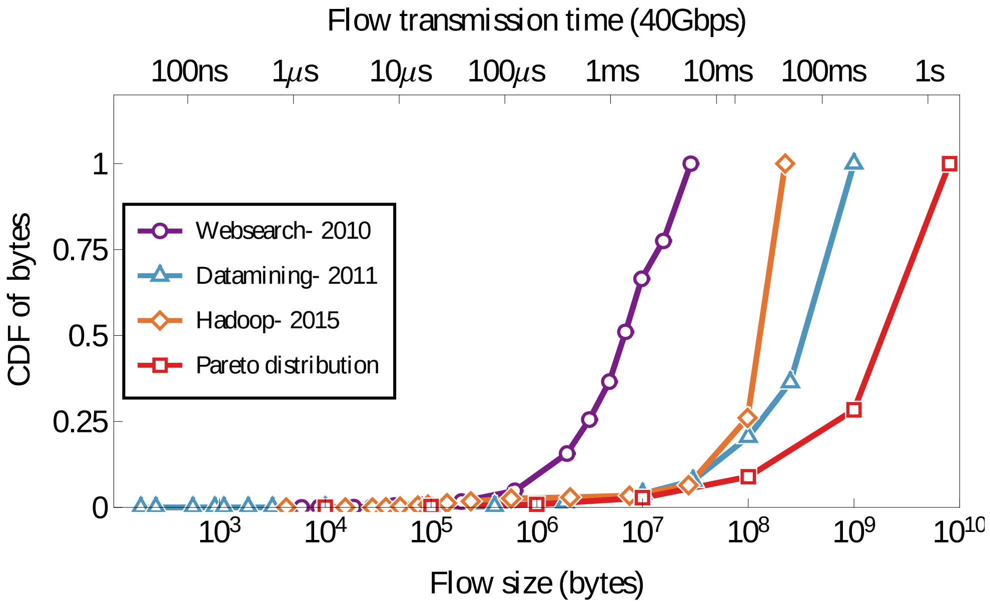
Flow Size Matters



⇢ **Observation 1:** Different apps have different flow size distributions.

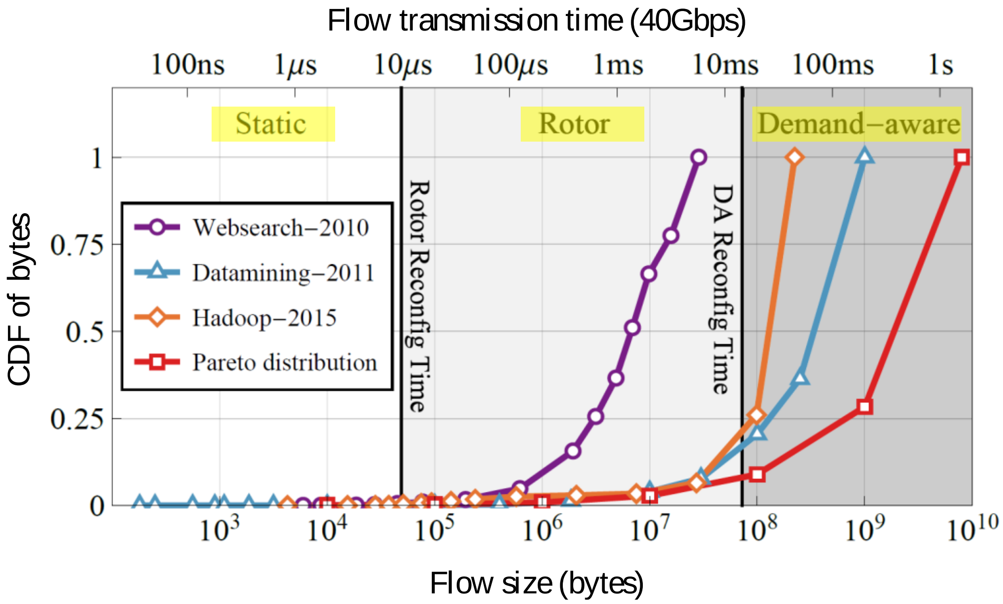
⇢ **Observation 2:** The transmission time of a flow depends on its size.

Flow Size Matters



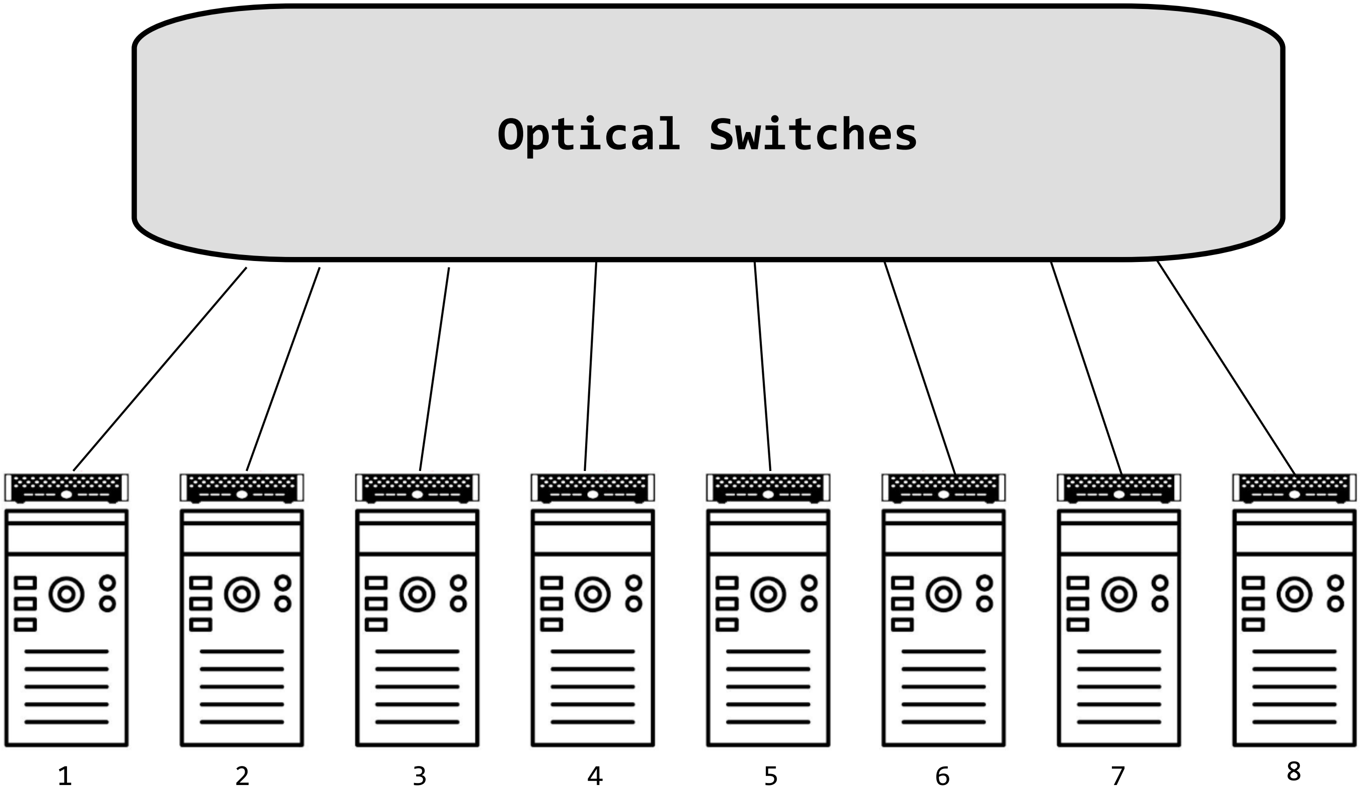
- **Observation 1:** Different apps have different flow size distributions.
- **Observation 2:** The transmission time of a flow depends on its size.
- **Observation 3:** For small flows, flow completion time suffers if network needs to be reconfigured first.
- **Observation 4:** For large flows, reconfiguration time may amortize.

Flow Size Matters

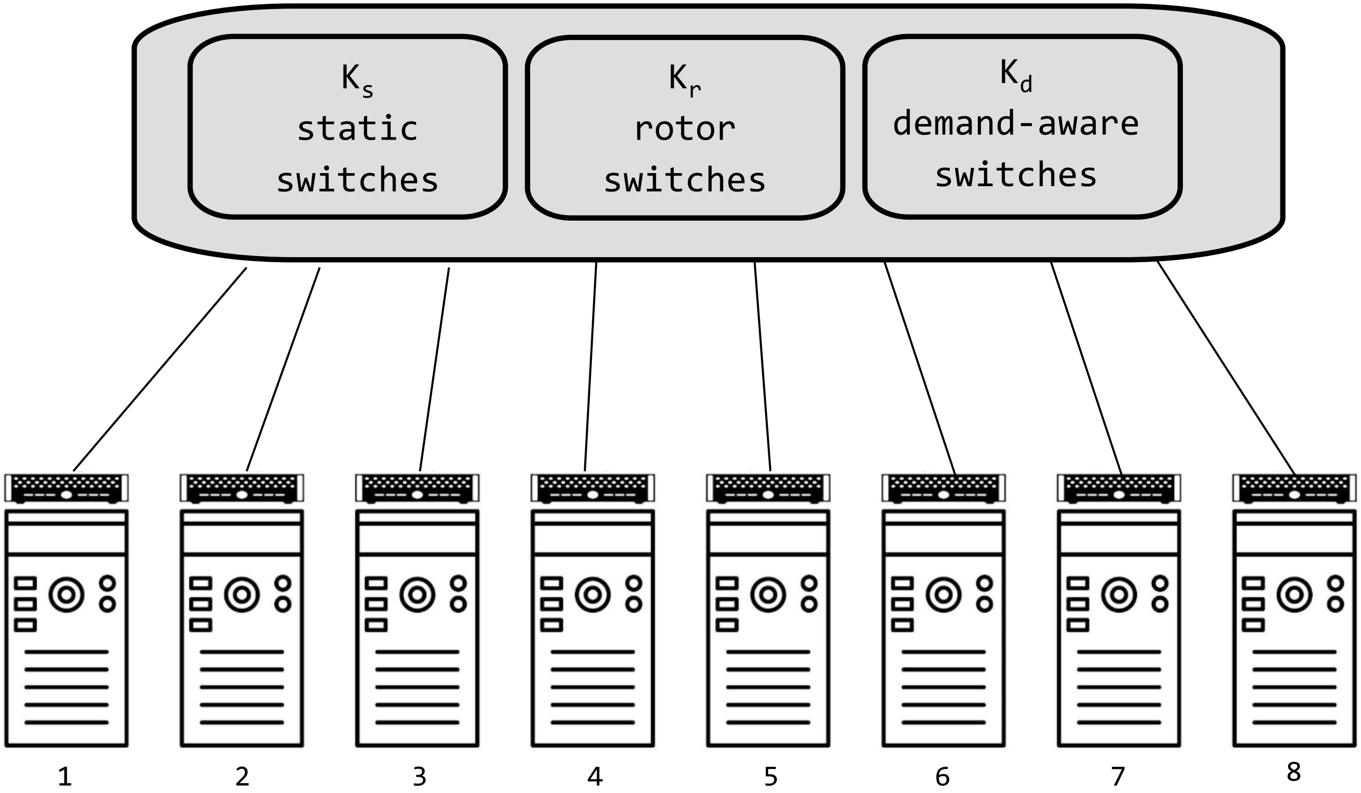


- **Observation 1:** Different apps have different flow size distributions.
- **Observation 2:** The transmission time of a flow depends on its size.
- **Observation 3:** For small flows, flow completion time suffers if network needs to be reconfigured first.
- **Observation 4:** For large flows, reconfiguration time may amortize.

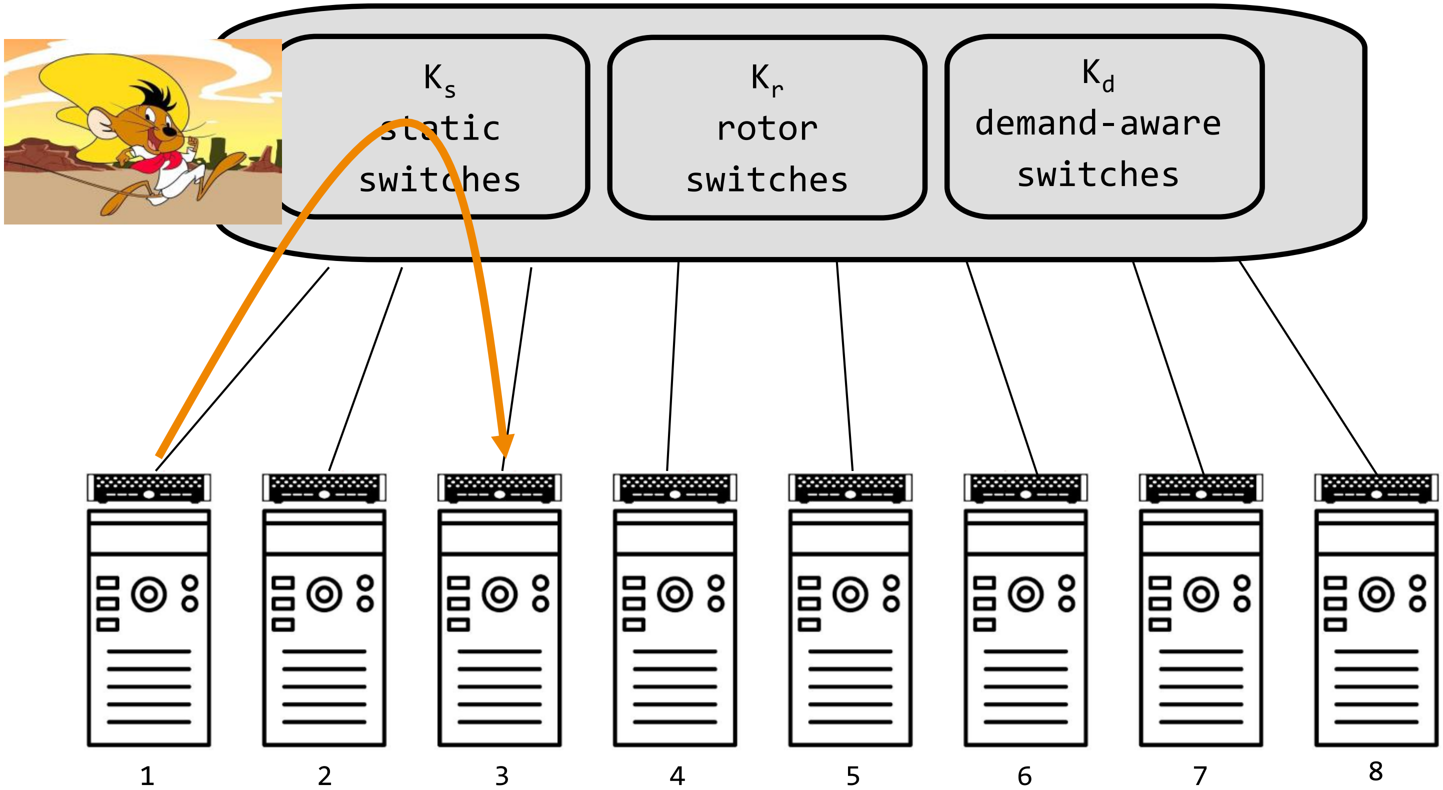
Cerberus



Cerberus

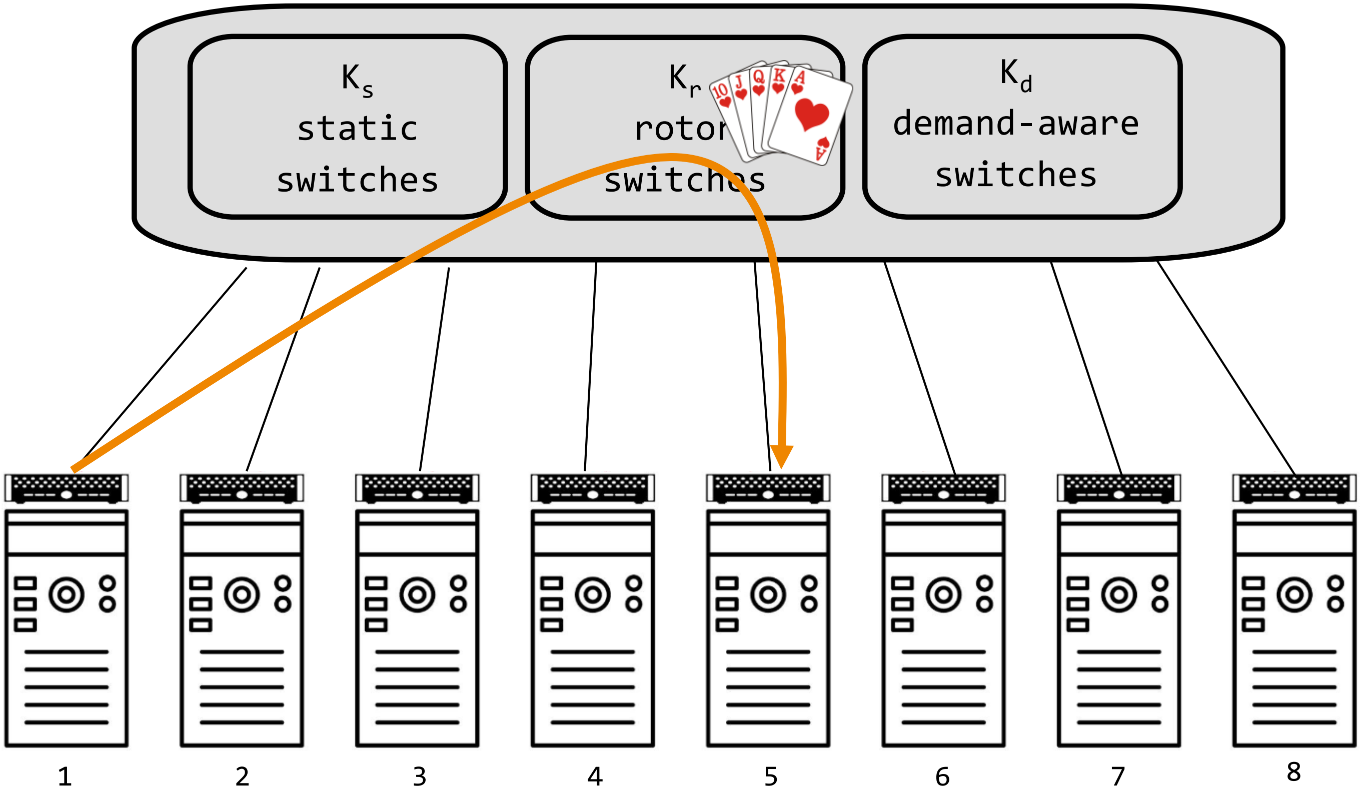

K_s
static
switches
K_r
rotor
switches
K_d
demand-aware
switches
1
2
3
4
5
6
7
8

Cerberus



Scheduling: Small flows go via static switches…

Cerberus



K_s
static
switches

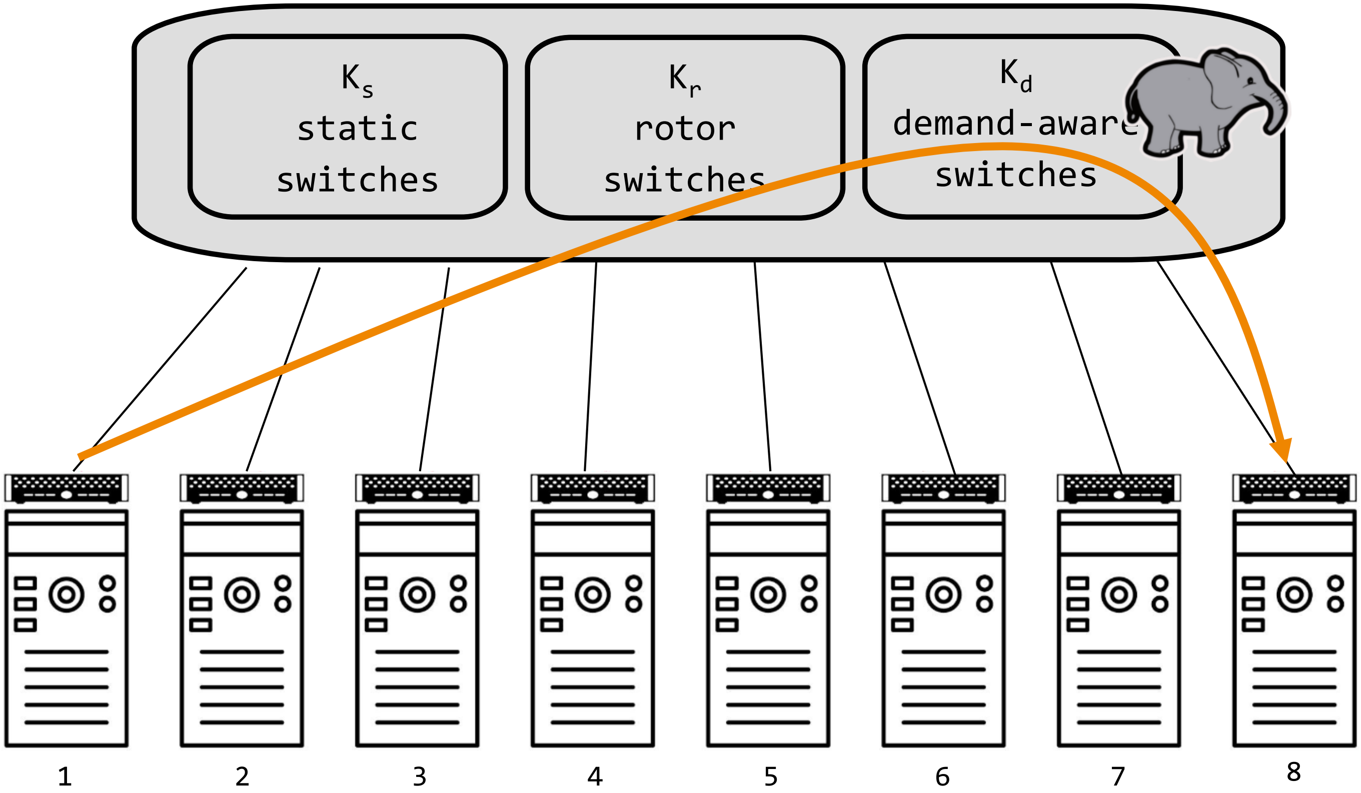
K_r
rotor
switches

K_d
demand-aware
switches

1 2 3 4 5 6 7 8

Scheduling: … medium flows via rotor switches…

Cerberus



Scheduling: … and large flows via demand-aware switches
(if one available, otherwise via rotor).

Excursion

More benefits of optical & reconfigurable switching

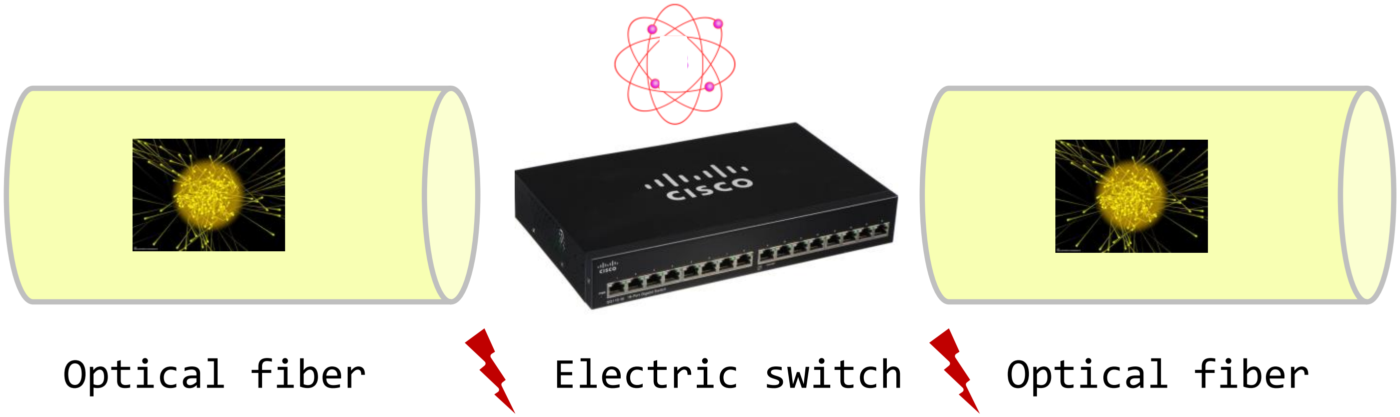
So far: focus on throughput performance.

Benefit 1:

Energy and Latency

- No need to *convert* photons in fiber to electrons in switch (and back)

- Can safe *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)

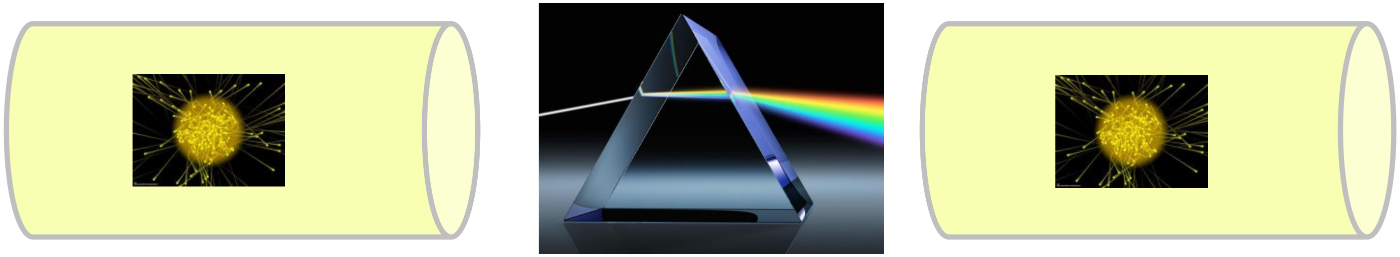


Benefit 1:

Energy and Latency

- No need to *convert* photons in fiber to electrons in switch (and back)

- Can safe *energy* and reduce *latency* (in addition to enabling almost unlimited throughput)

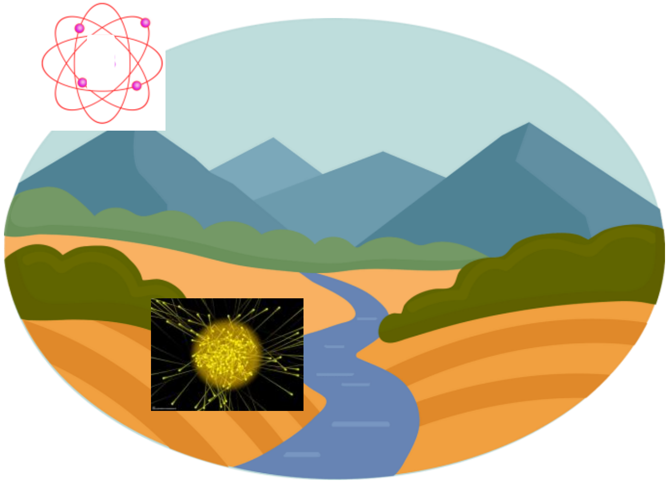


Optical fiber —— Optical switch —— Optical fiber

Benefit 2:

Resilience

Floodings in South Germany destroyed much electrical network infrastructure



Solution: deploy optical infrastructure (in valleys) and electrical *on hills* where safe?

Benefit 3:

Evolving Datacenters

- ⇢ Reconfigurable datacenter networks naturally support *heterogeneous* network elements
- ⇢ And therefore also *incremental* hardware upgrades

Amin Vahdat
Google

Systems

Jupiter evolving: Reflecting on Google's data center network transformation

August 24, 2022

Google Cloud

Conclusion

- Opportunity: *structure* in demand and *reconfigurable* networks
- So far: tip of the iceberg
- Many challenges
  - Optimal design depends on traffic pattern
  - How to *measure/predict* traffic?
  - Impact on other *layers*?
  - Routing and congestion control?
  - *Scalable control* plane
  - *Application-specific* self-adjusting networks?
- Many more *opportunities* for optical networks

More Details: Interivews

We recently interviewed three experts

Amin Vahdat
Google

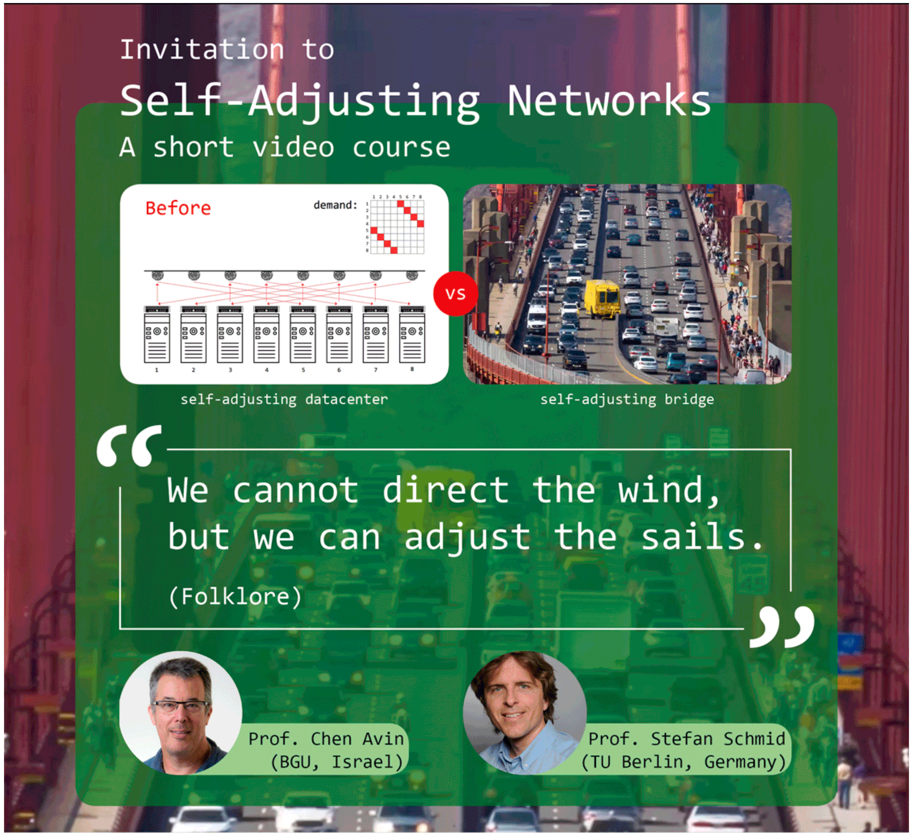
Manya Ghobadi
MIT

George Papen
UCSD

“Think about a machine learning training job, say, training a ***ChatGPT*** model. It takes months to train this model, but the traffic matrix is beautifully ***predictable and periodic***, which makes it very suitable to think about whether or not we could ***adjust the topology*** according to the traffic.” –Manya Gobhadi (MIT)

Watch here:
https://www.youtube.com/@self-adjusting-networks-course

Online Video Course



Invitation to

Self-Adjusting Networks

A short video course

Before
demand:
vs

self-adjusting datacenter

self-adjusting bridge

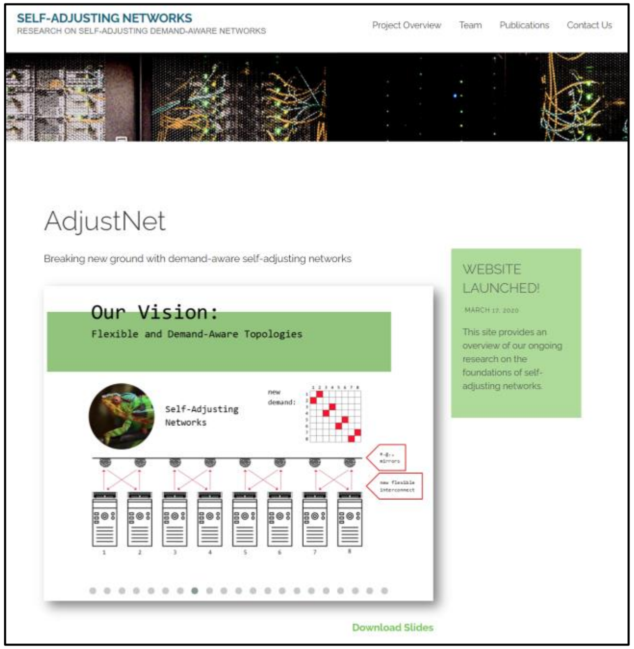
> “We cannot direct the wind,
> but we can adjust the sails.
>
> (Folklore)”

Prof. Chen Avin
(BGU, Israel)

Prof. Stefan Schmid
(TU Berlin, Germany)

https://self-adjusting.net/course

Websites



SELF-ADJUSTING NETWORKS
RESEARCH ON SELF-ADJUSTING DEMAND-AWARE NETWORKS

Project Overview Team Publications Contact Us

AdjustNet

Breaking new ground with demand-aware self-adjusting networks

Our Vision:
Flexible and Demand-Aware Topologies

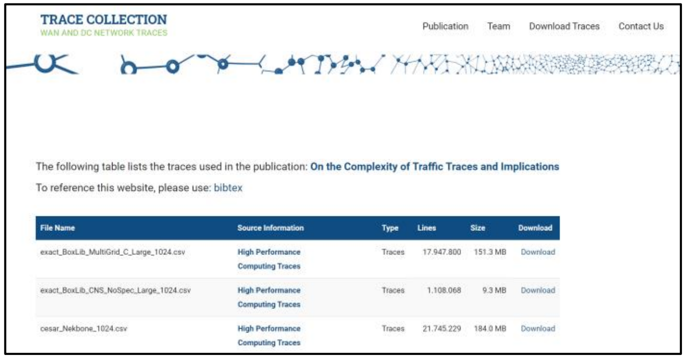
WEBSITE LAUNCHED!

MARCH 17, 2020

This site provides an overview of our ongoing research on the foundations of self-adjusting networks.

Download Slides

http://self-adjusting.net/
Project website



TRACE COLLECTION
WAN AND DC NETWORK TRACES

Publication Team Download Traces Contact Us

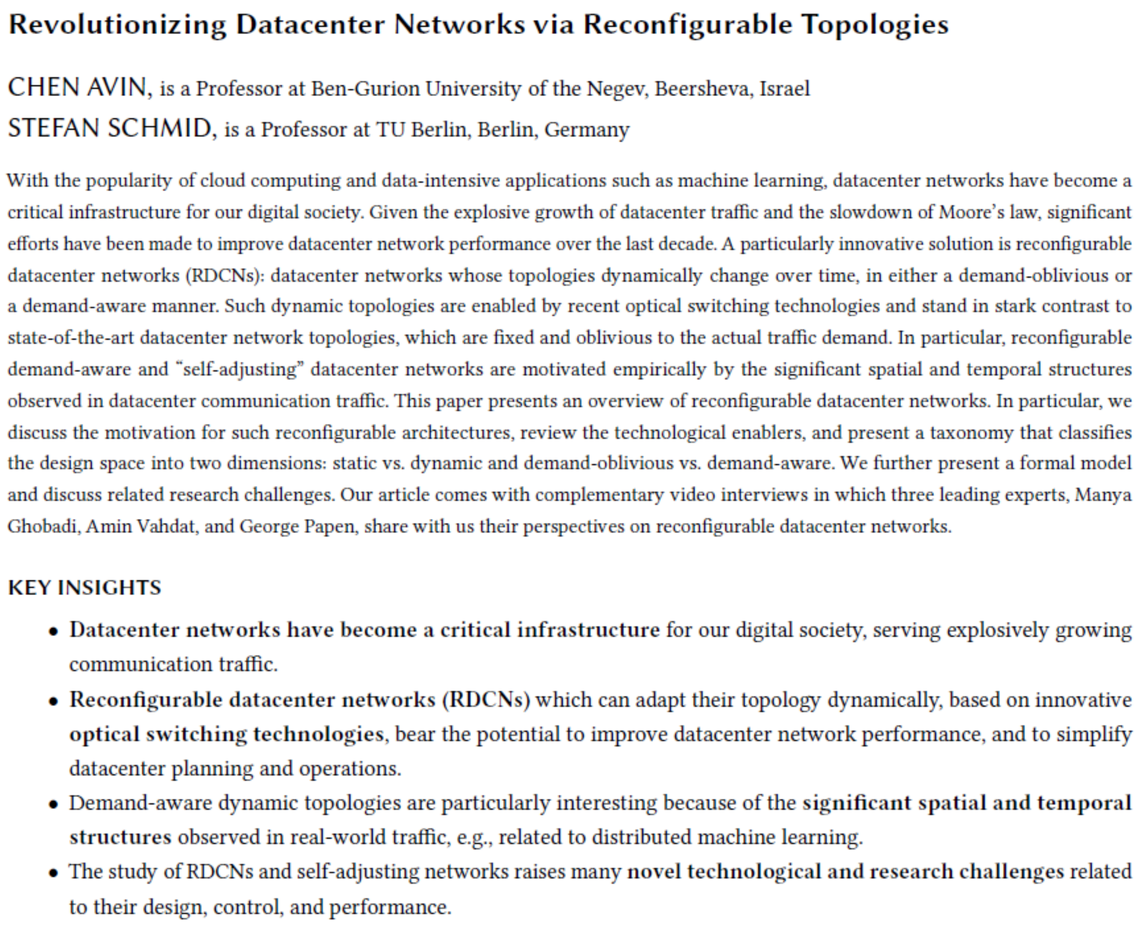
The following table lists the traces used in the publication: **On the Complexity of Traffic Traces and Implications**

To reference this website, please use: bibtex

File Name	Source Information	Type	Lines	Size	Download
exact_BoxLib_MultiGrid_C_Large_1024.csv	**High Performance Computing Traces**	Traces	17.947.800	151.3 MB	Download
exact_BoxLib_CNS_NoSpec_Large_1024.csv	**High Performance Computing Traces**	Traces	1.108.068	9.3 MB	Download
cesar_Nekbone_1024.csv	**High Performance Computing Traces**	Traces	21.745.229	184.0 MB	Download

https://trace-collection.net/
Trace collection website

Upcoming CACM Article



Revolutionizing Datacenter Networks via Reconfigurable Topologies

CHEN AVIN, is a Professor at Ben-Gurion University of the Negev, Beersheva, Israel

STEFAN SCHMID, is a Professor at TU Berlin, Berlin, Germany

With the popularity of cloud computing and data-intensive applications such as machine learning, datacenter networks have become a critical infrastructure for our digital society. Given the explosive growth of datacenter traffic and the slowdown of Moore's law, significant efforts have been made to improve datacenter network performance over the last decade. A particularly innovative solution is reconfigurable datacenter networks (RDCNs): datacenter networks whose topologies dynamically change over time, in either a demand-oblivious or a demand-aware manner. Such dynamic topologies are enabled by recent optical switching technologies and stand in stark contrast to state-of-the-art datacenter network topologies, which are fixed and oblivious to the actual traffic demand. In particular, reconfigurable demand-aware and "self-adjusting" datacenter networks are motivated empirically by the significant spatial and temporal structures observed in datacenter communication traffic. This paper presents an overview of reconfigurable datacenter networks. In particular, we discuss the motivation for such reconfigurable architectures, review the technological enablers, and present a taxonomy that classifies the design space into two dimensions: static vs. dynamic and demand-oblivious vs. demand-aware. We further present a formal model and discuss related research challenges. Our article comes with complementary video interviews in which three leading experts, Manya Ghobadi, Amin Vahdat, and George Papen, share with us their perspectives on reconfigurable datacenter networks.

KEY INSIGHTS

- **Datacenter networks have become a critical infrastructure** for our digital society, serving explosively growing communication traffic.
- **Reconfigurable datacenter networks (RDCNs)** which can adapt their topology dynamically, based on innovative **optical switching technologies**, bear the potential to improve datacenter network performance, and to simplify datacenter planning and operations.
- Demand-aware dynamic topologies are particularly interesting because of the **significant spatial and temporal structures** observed in real-world traffic, e.g., related to distributed machine learning.
- The study of RDCNs and self-adjusting networks raises many **novel technological and research challenges** related to their design, control, and performance.

More References

Mars: Near-Optimal Throughput with Shallow Buffers in Reconfigurable Datacenter Networks
Vamsi Addanki, Chen Avin, and Stefan Schmid.
ACM **SIGMETRICS** and ACM Performance Evaluation Review (**PER**), Orlando, Florida, USA, June 2023.

Duo: A High-Throughput Reconfigurable Datacenter Network Using Local Routing and Control
Johannes Zerwas, Csaba Györgyi, Andreas Blenk, Stefan Schmid, and Chen Avin.
ACM **SIGMETRICS** and ACM Performance Evaluation Review (**PER**), Orlando, Florida, USA, June 2023.

Cerberus: The Power of Choices in Datacenter Topology Design (A Throughput Perspective)
Chen Griner, Johannes Zerwas, Andreas Blenk, Manya Ghobadi, Stefan Schmid, and Chen Avin.
ACM **SIGMETRICS** and ACM Performance Evaluation Review (**PER**), Mumbai, India, June 2022.

Demand-Aware Network Design with Minimal Congestion and Route Lengths
Chen Avin, Kaushik Mondal, and Stefan Schmid.
IEEE/ACM Transactions on Networking (**TON**), 2022.

On the Complexity of Traffic Traces and Implications
Chen Avin, Manya Ghobadi, Chen Griner, and Stefan Schmid.
ACM **SIGMETRICS** and ACM Performance Evaluation Review (**PER**), Boston, Massachusetts, USA, June 2020

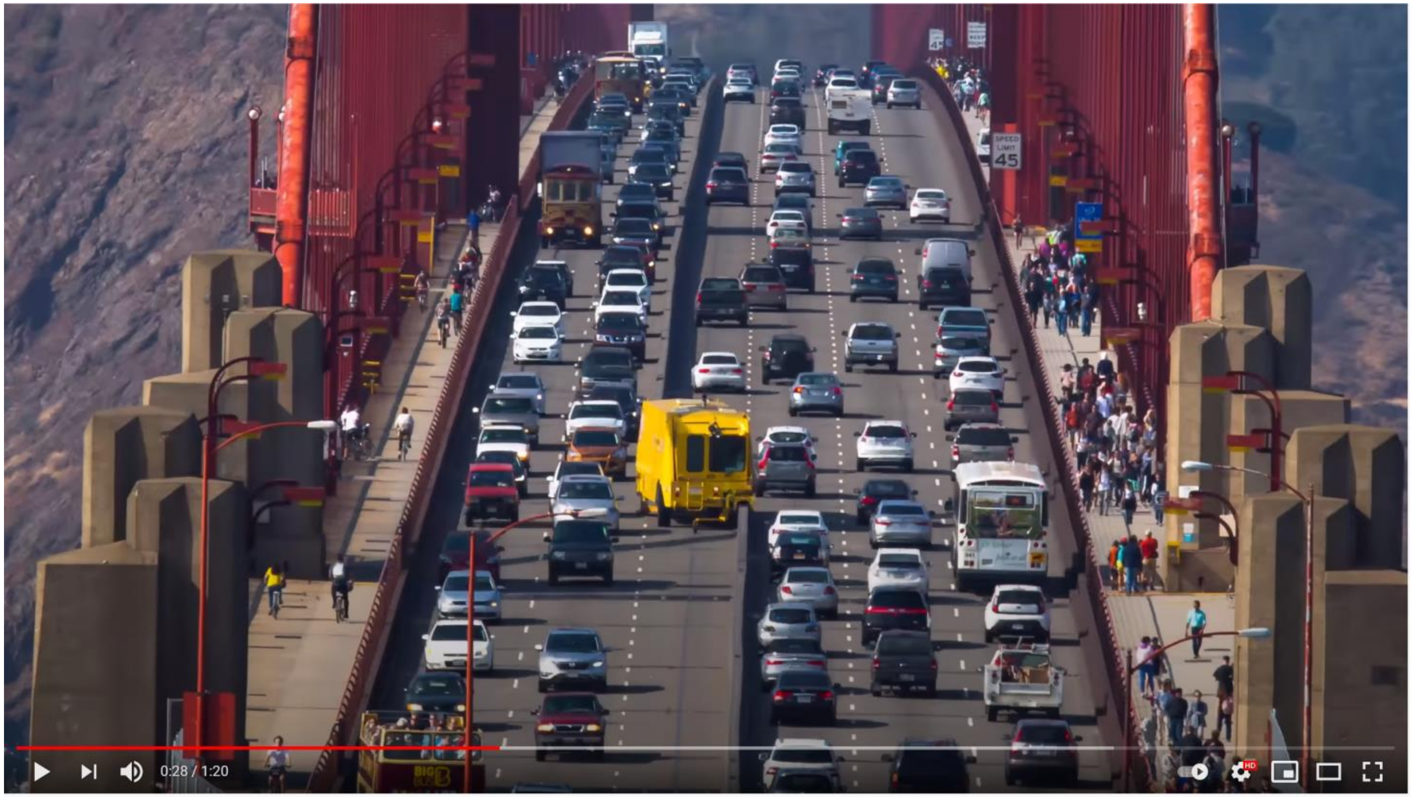
A Survey of Reconfigurable Optical Networks
Matthew Nance Hall, Klaus-Tycho Foerster, Stefan Schmid, and Ramakrishnan Durairajan.
Optical Switching and Networking (**OSN**), Elsevier, 2021.

Toward Demand-Aware Networking: A Theory for Self-Adjusting Networks (Editorial)
Chen Avin and Stefan Schmid.
ACM SIGCOMM Computer Communication Review (**CCR**), October 2018.

SplayNet: Towards Locally Self-Adjusting Networks
Stefan Schmid, Chen Avin, Christian Scheideler, Michael Borokhovich, Bernhard Haeupler, and Zvi Lotker.
IEEE/ACM Transactions on Networking (**TON**), Volume 24, Issue 3, 2016.

.

Questions?



Slides
available
here:

Bonus Material

Hogwarts Stair

Question:

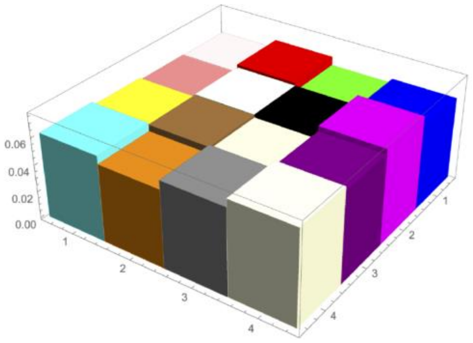
How to Quantify such “Structure” in the Demand?

Intuition

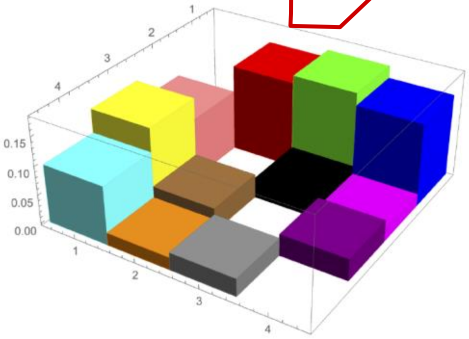
Which demand has more structure?

⇢ Traffic matrices of two different distributed ML applications

- → GPU-to-GPU



VS



Color = communication pair

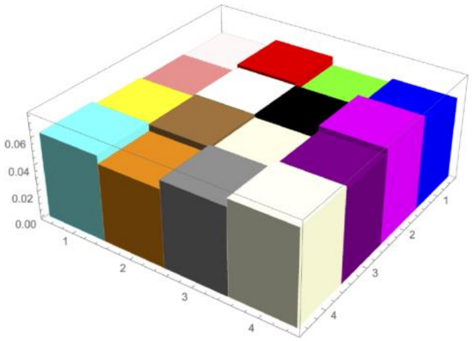
Intuition

Which demand has more structure?

⇢ Traffic matrices of two different distributed ML applications

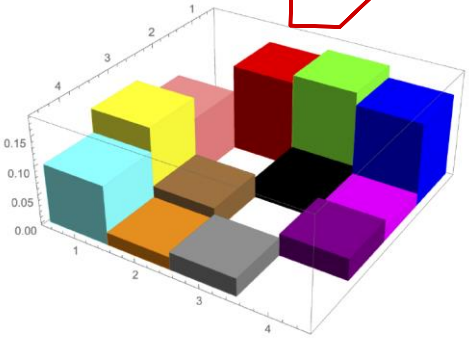
- ⇀ GPU-to-GPU

Color = communication pair



More uniform

VS



More structure

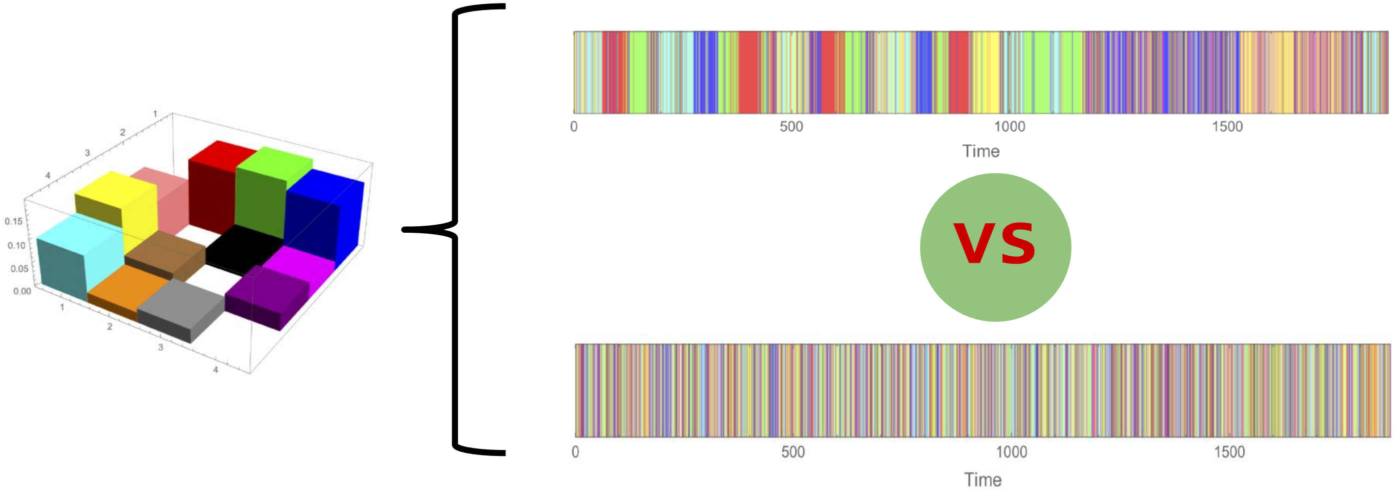
Intuition

Spatial vs temporal structure

⇢ Two different ways to generate same traffic matrix:

⇀ Same non-temporal structure

⇢ Which one has more structure?



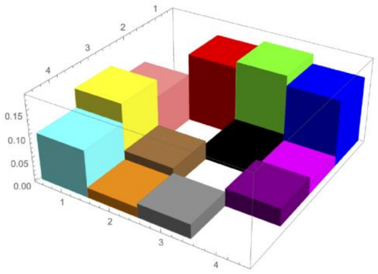
Intuition

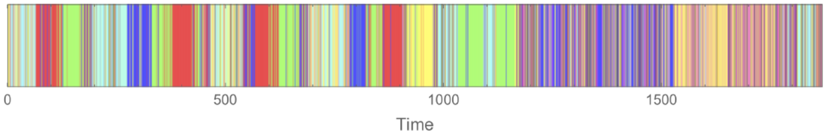
Spatial vs temporal structure

⇢ Two different ways to generate same traffic matrix:

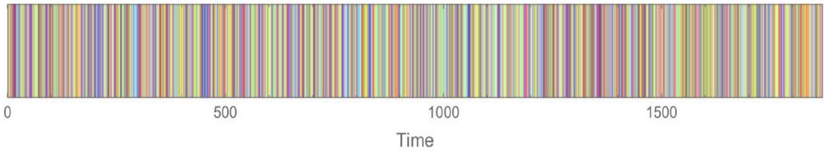
→ Same non-temporal structure

⇢ Which one has more structure?





VS

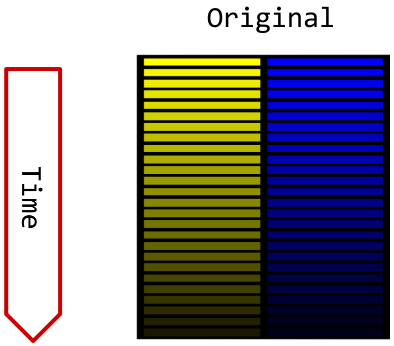


Systematically?

Trace Complexity

Information-Theoretic Approach

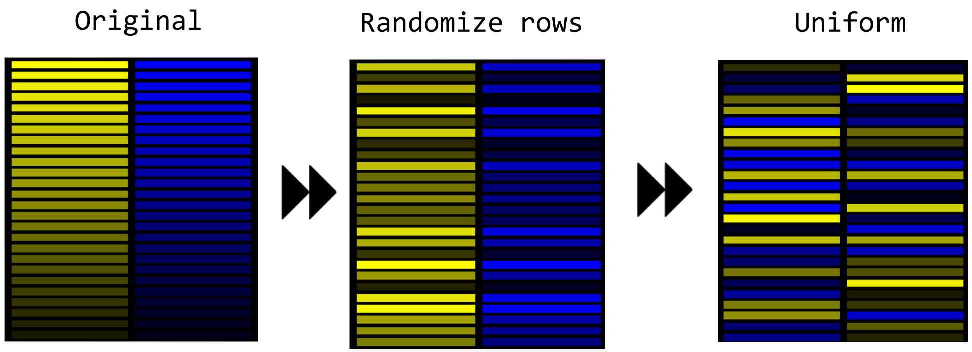
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

“Shuffle&Compress”



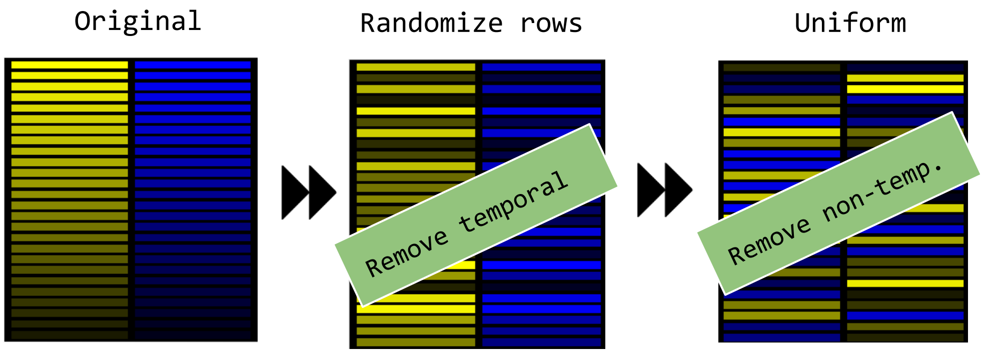
Increasing complexity (systematically randomized)

More structure (compresses better)

Trace Complexity

Information-Theoretic Approach

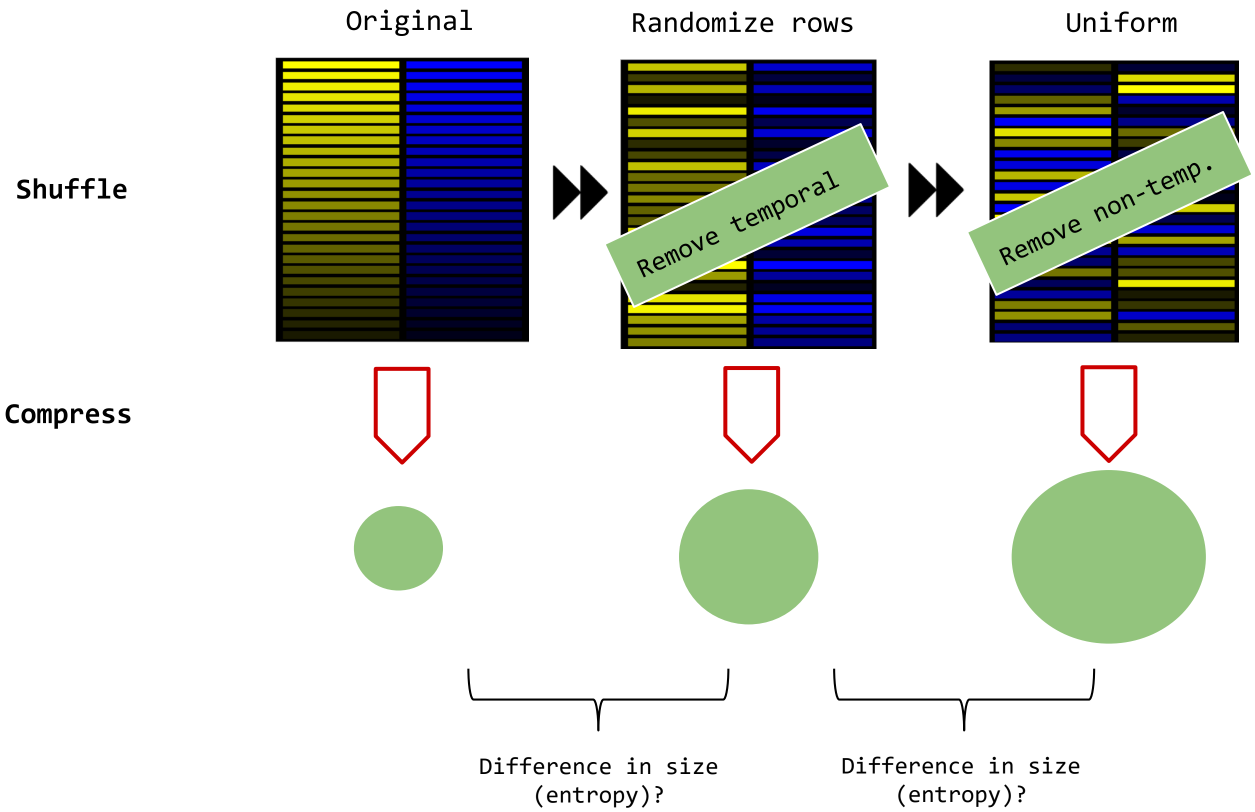
“Shuffle&Compress”



Trace Complexity

Information-Theoretic Approach

“Shuffle&Compress”



Original

Randomize rows

Uniform

Shuffle

Remove temporal

Remove non-temp.

Compress

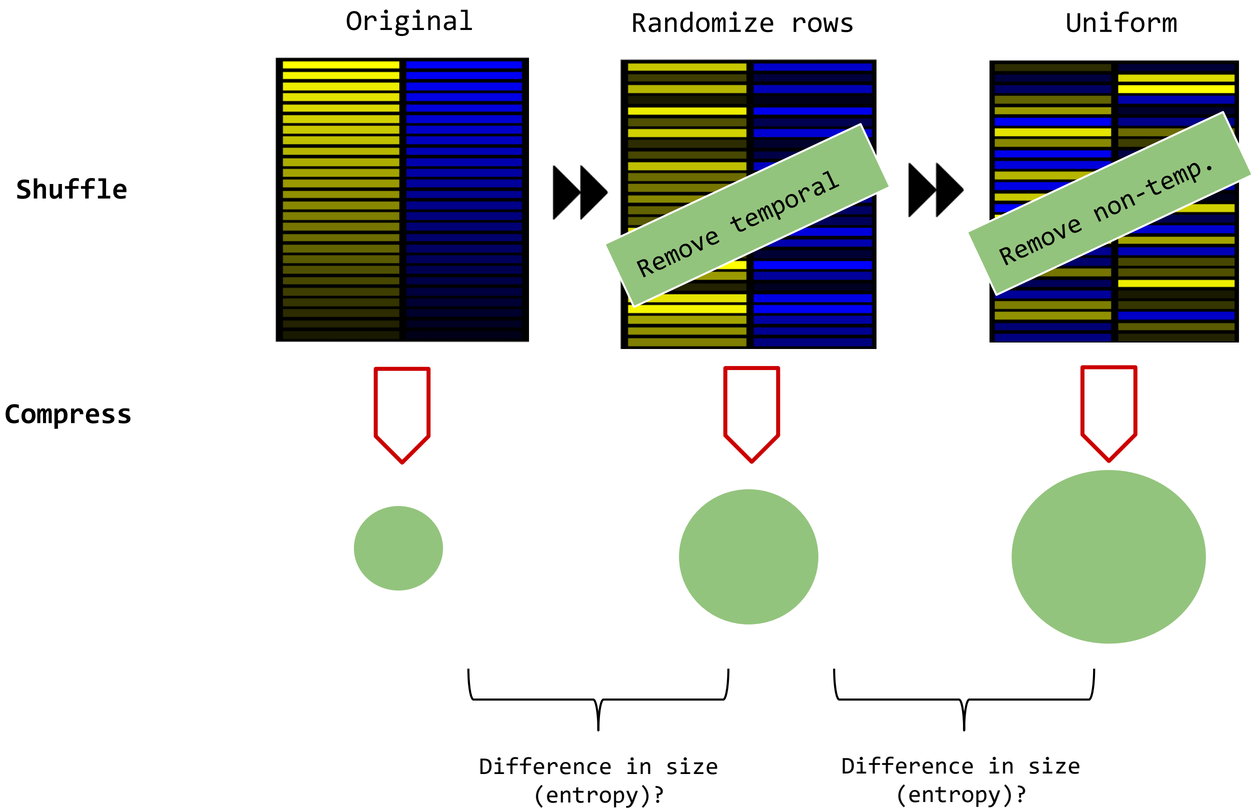
Difference in size
(entropy)?

Difference in size
(entropy)?

Trace Complexity

Information-Theoretic Approach

“Shuffle&Compress”



Original

Randomize rows

Uniform

Shuffle

Compress

Remove temporal

Remove non-temp.

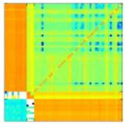
Difference in size (entropy)?

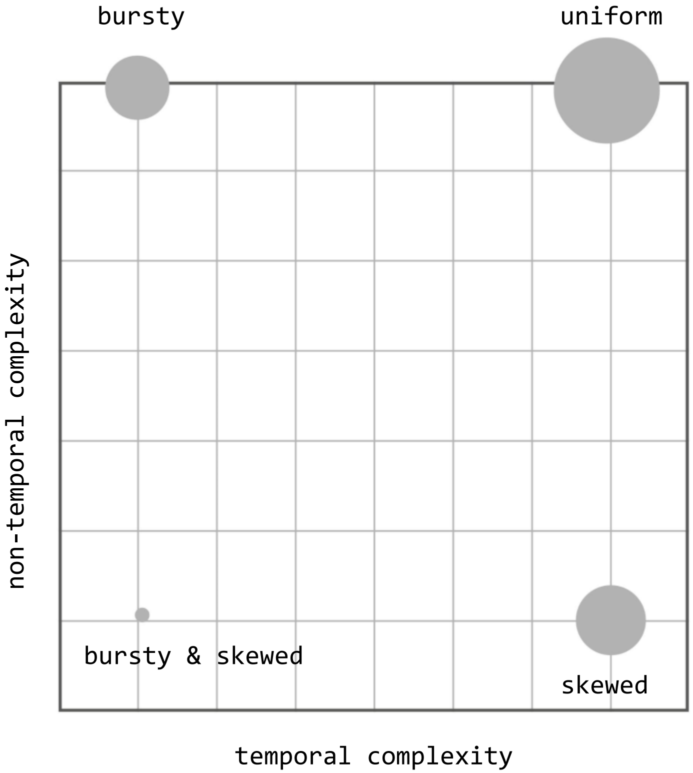
Difference in size (entropy)?

Can be used to define 2-dimensional **complexity map!**

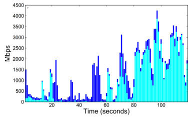
Our Methodology

Complexity Map





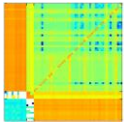
No structure

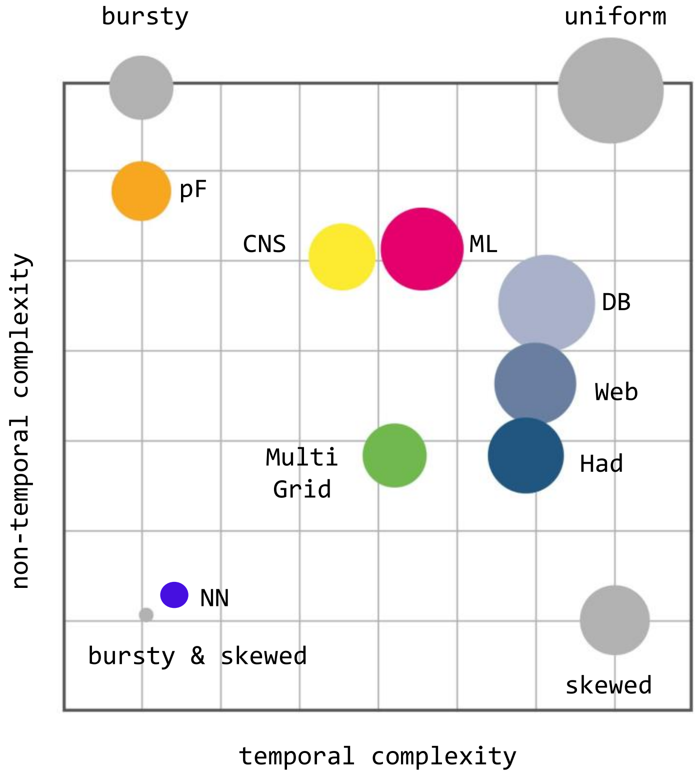


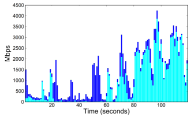
Our **approach**: iterative **randomization and compression** of trace to identify dimensions of structure.

Our Methodology

Complexity Map







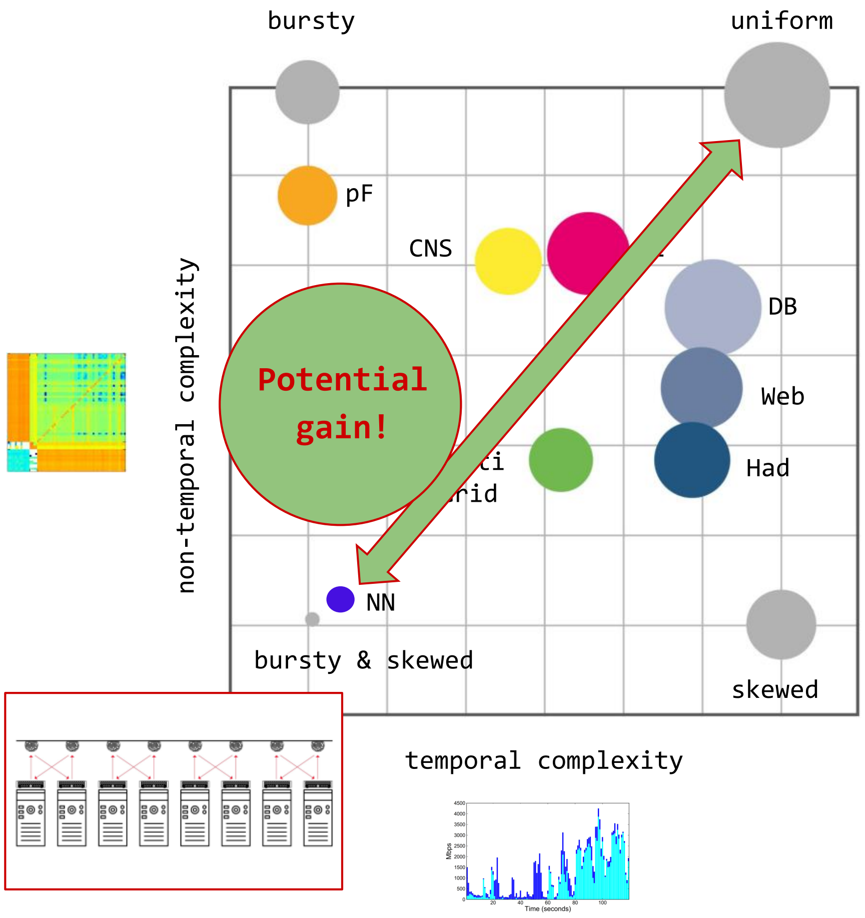
No structure

Our **approach**: iterative **randomization and compression** of trace to identify dimensions of structure.

Different structures!

Our Methodology

Complexity Map



Potential gain!

Our **approach**: iterative **randomization and compression** of trace to identify dimensions of structure.

Different structures!

Further Reading

ACM SIGMETRICS 2020

On the Complexity of Traffic Traces and Implications

CHEN AVIN, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel
MANYA GHOBADI, Computer Science and Artificial Intelligence Laboratory, MIT, USA
CHEN GRINER, School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel
STEFAN SCHMID, Faculty of Computer Science, University of Vienna, Austria

This paper presents a systematic approach to identify and quantify the types of structures featured by packet traces in communication networks. Our approach leverages an information-theoretic methodology, based on iterative randomization and compression of the packet trace, which allows us to systematically remove and measure dimensions of structure in the trace. In particular, we introduce the notion of *trace complexity* which approximates the entropy rate of a packet trace. Considering several real-world traces, we show that trace complexity can provide unique insights into the characteristics of various applications. Based on our approach, we also propose a traffic generator model able to produce a synthetic trace that matches the complexity levels of its corresponding real-world trace. Using a case study in the context of datacenters, we show that insights into the structure of packet traces can lead to improved demand-aware network designs: datacenter topologies that are optimized for specific traffic patterns.

CCS Concepts: • **Networks** → **Network performance evaluation**; **Network algorithms**; **Data center networks**; • **Mathematics of computing** → *Information theory*;

Additional Key Words and Phrases: trace complexity, self-adjusting networks, entropy rate, compress, complexity map, data centers

ACM Reference Format:
Chen Avin, Manya Ghobadi, Chen Griner, and Stefan Schmid. 2020. On the Complexity of Traffic Traces and Implications. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 1, Article 20 (March 2020), 29 pages. https://doi.org/10.1145/3379486

20

1 INTRODUCTION

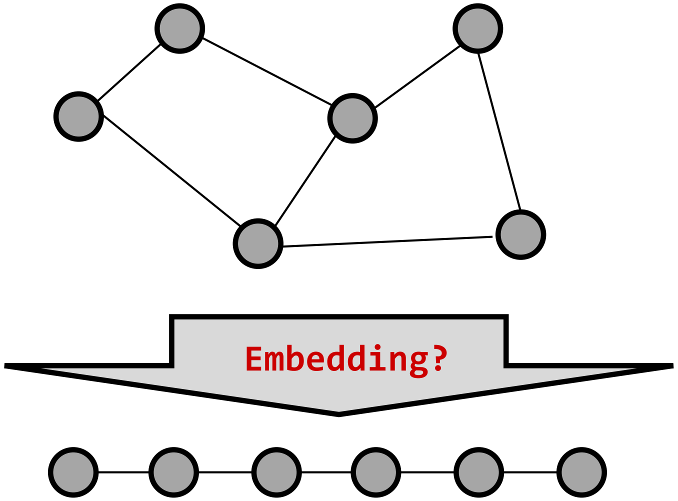
Packet traces collected from networking applications, such as datacenter traffic, have been shown to feature much *structure*: datacenter traffic matrices are sparse and skewed [16, 39], exhibit

Related Problem: Remember Bernardetta’s Talk

Virtual Network Embedding Problem (VNEP)

Example △=2: A Minium Linear Arrangement (**MLA**) Problem

→ Minimizes sum of virtual edges

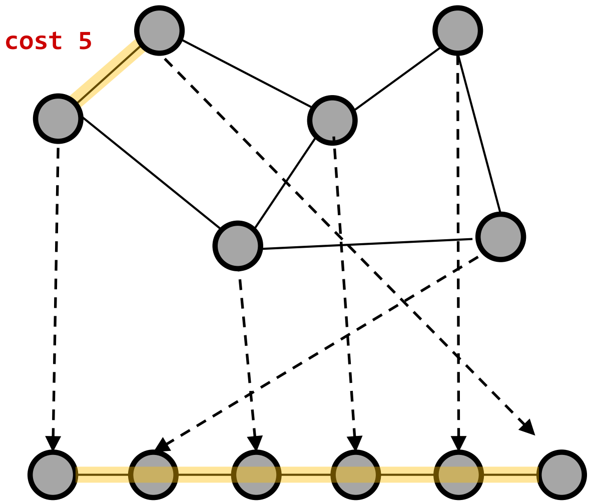


Related Problem: Remember Bernardetta’s Talk

Virtual Network Embedding Problem (VNEP)

Example △=2: A Minium Linear Arrangement (**MLA**) Problem

→ Minimizes sum of virtual edges



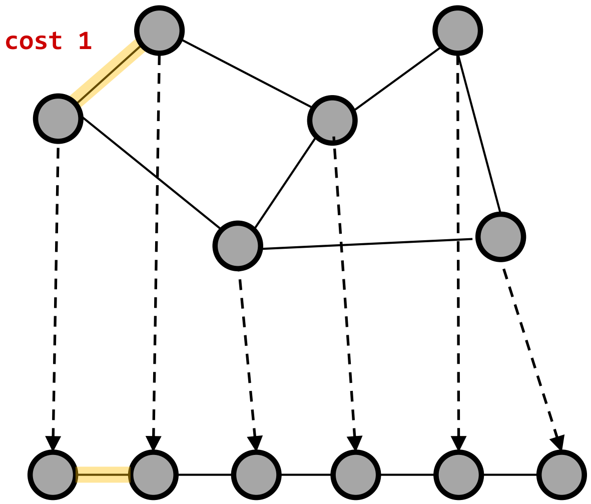
Bad!

Related Problem: Remember Bernardetta’s Talk

Virtual Network Embedding Problem (VNEP)

Example △=2: A Minium Linear Arrangement (**MLA**) Problem

→ Minimizes sum of virtual edges



Good!

Related Problem: Remember Bernardetta’s Talk

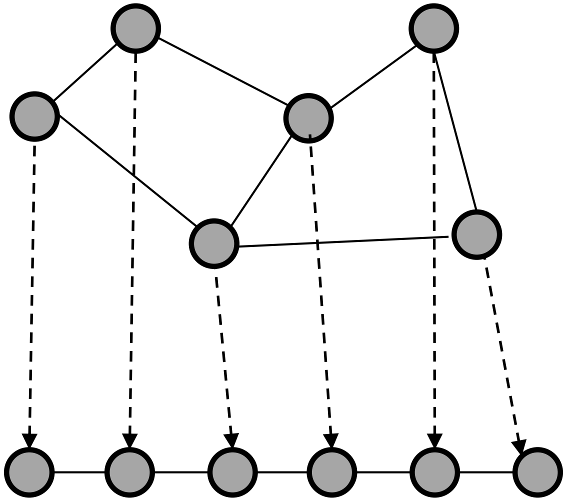
Virtual Network Embedding Problem (VNEP)

Example △=2: A Minium Linear Arrangement (**MLA**) Problem

⇀ Minimizes sum of virtual edges

MLA is **NP-hard**

⇀ … and so is our problem!



Related Problem: Remember Bernardetta’s Talk

Virtual Network Embedding Problem (VNEP)

Example $\triangle$=2: A Minium Linear Arrangement (**MLA**) Problem

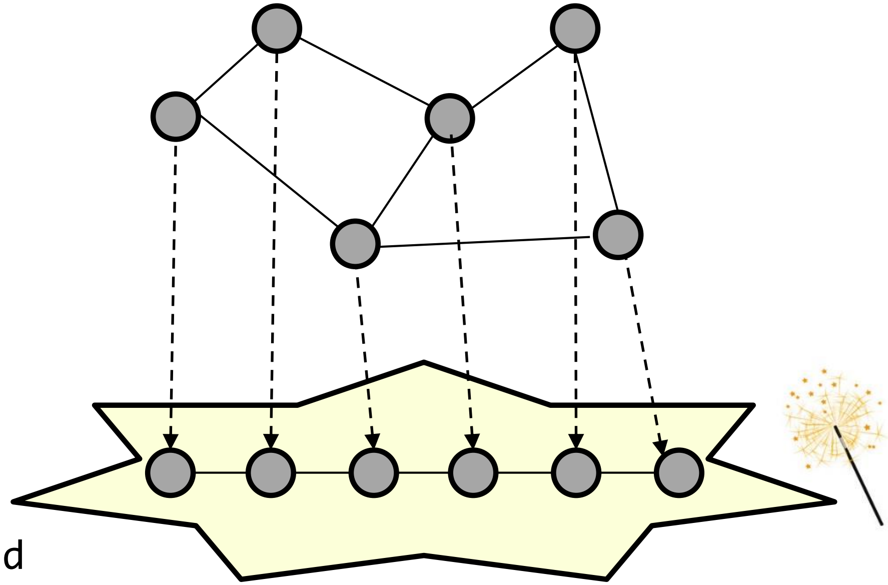
- ⇀ Minimizes sum of virtual edges

MLA is **NP-hard**

- ⇀ … and so is our problem!

But what about **$\triangle$>2**?

- ⇀ Embedding problem still hard
- ⇀ But we have a new **degree of freedom**!



Related Problem: Remember Bernardetta’s Talk

Virtual Network Embedding Problem (VNEP)

Example △=2: A Minium Linear Arrangement (**MLA**) Problem

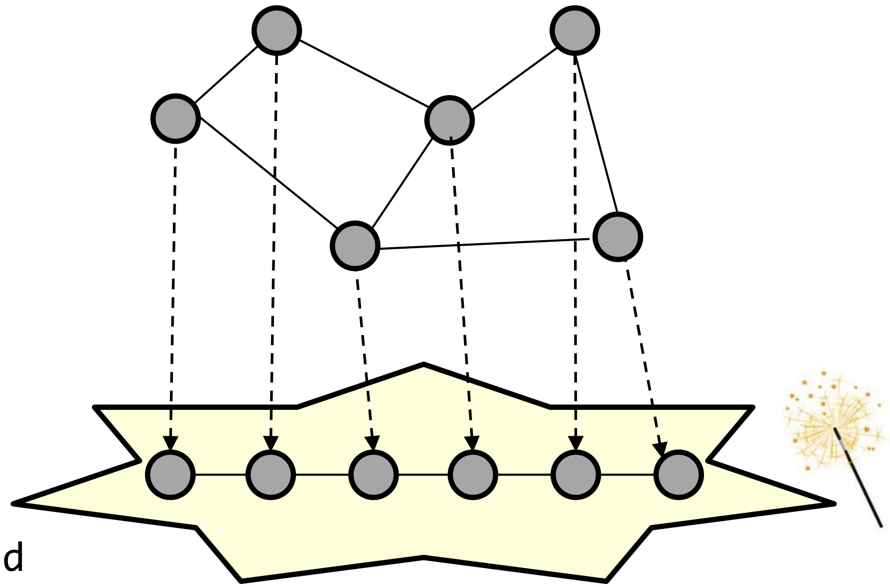
- Minimizes sum of virtual edges

MLA is **NP-hard**

- … and so is our problem!

But what about **△>2**?

- Embedding problem still hard
- But we have a new **degree of freedom**!



Simplifies problem?!

Another Related Problem

Low Distortion Spanners

- Classic problem: find *sparse*, *distance-preserving* (low-distortion) spanner of a graph

- But:
  - Spanners aim at low distortion *among all pairs*; in our case, we are only interested in the local distortion, 1-hop communication neighbors
  - We allow *auxiliary edges* (not a subgraph): similar to geometric spanners
  - We require *constant degree*

From Spanners to DANs

An Algorithm

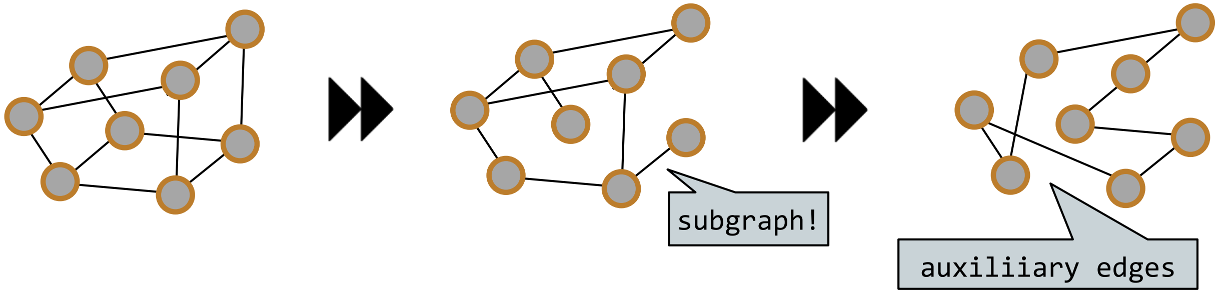
⇢ Yet, can leverage the connection to spanners sometimes!

Theorem: If demand matrix is regular and uniform, and if we can find a constant distortion, linear sized (i.e., constant, sparse) spanner for this request graph: then we can design a constant degree DAN providing an optimal expected route length *(i.e., $O(H(X|Y)+H(Y|X))$).*

r-regular *and* ***uniform*** **demand***:*

Sparse, ***irregular*** ***(constant)*** **spanner***:*

Constant *degree* ***optimal*** **DAN** (ERL at most ***log r***):



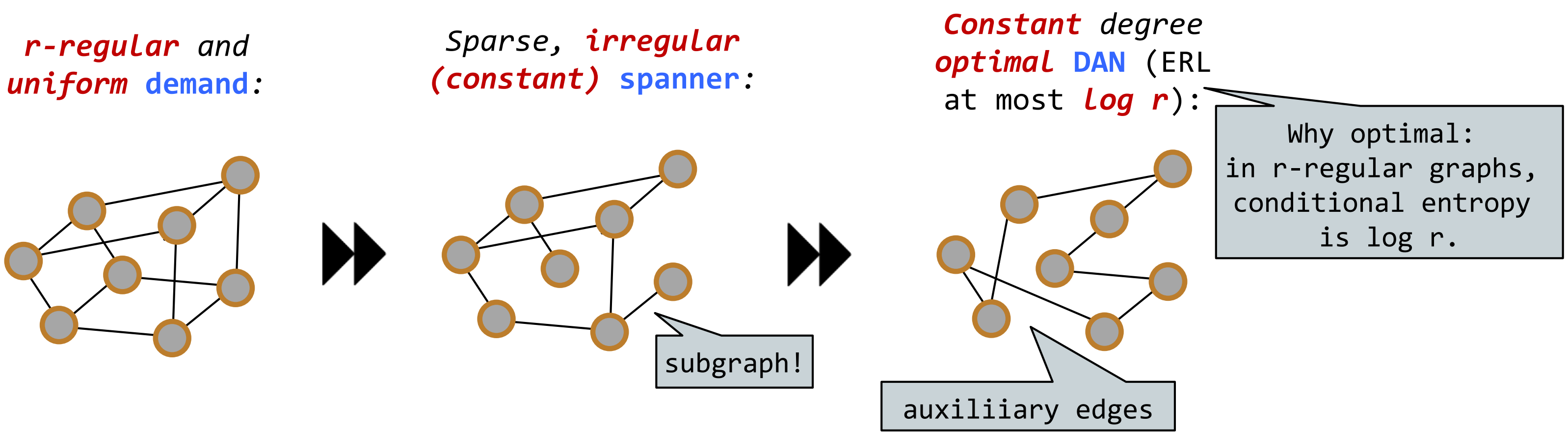
From Spanners to DANs

An Algorithm

⇢ Yet, can leverage the connection to spanners sometimes!

Theorem: If demand matrix is regular and uniform, and if we can find a constant distortion, linear sized (i.e., constant, sparse) spanner for this request graph: then we can design a constant degree DAN providing an optimal expected route length *(i.e., $O(H(X|Y)+H(Y|X))$.*

Our degree reduction trick again!



r-regular *and* ***uniform demand****:*

Sparse, ***irregular (constant) spanner****:*

Constant *degree* ***optimal*** **DAN** (ERL at most ***log r***):

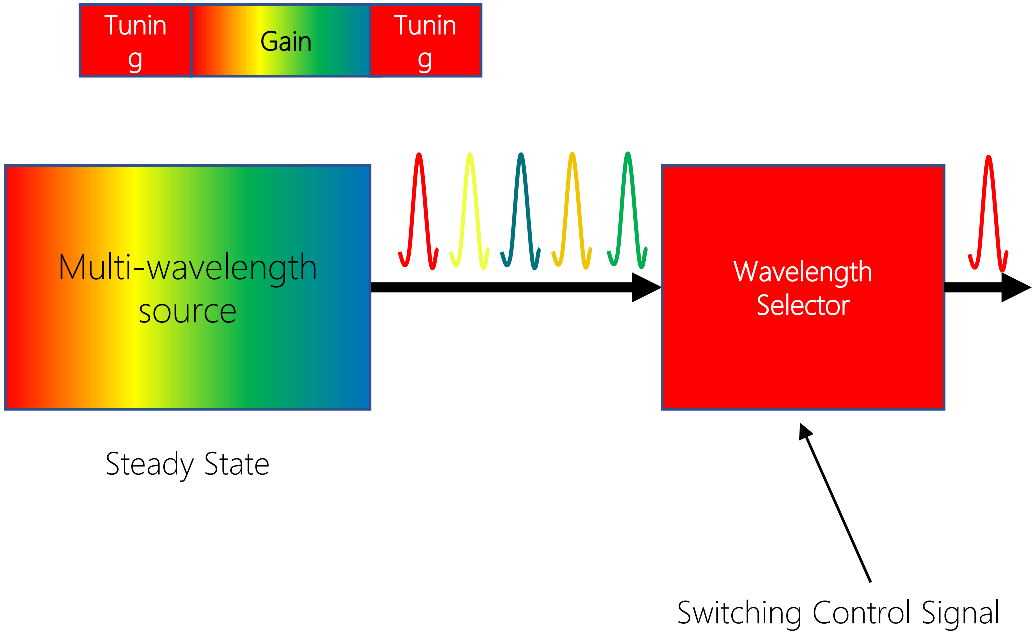
Why optimal: in r-regular graphs, conditional entropy is log r.

subgraph!

auxiliiary edges

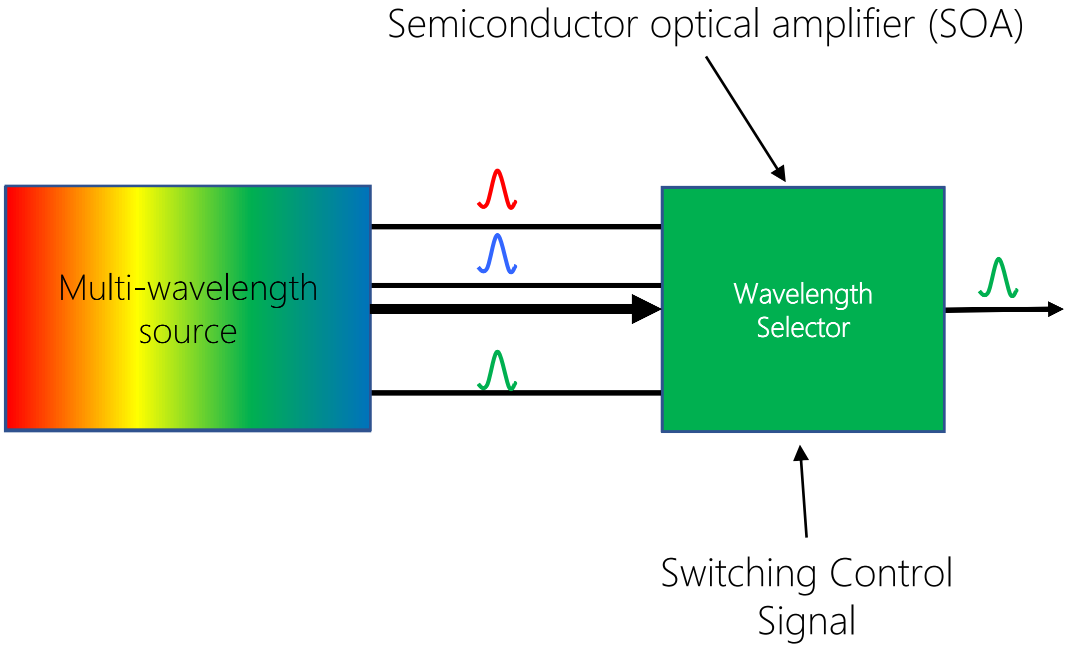
Idea

Disaggregated Laser



Ballani et al., Sirius, ACM SIGCOMM 2020.

Example Design



Sirius also implemented other designs
(details in the paper)

Ballani et al., Sirius, ACM SIGCOMM 2020.